Computabilidad y logica

May 6, 2023

Contents

1 Notacion y conceptos basicos

Usaremos **R** para denotar el conjunto de los numeros reales, **Z** para denotar el conjunto de los numeros enteros, **N** para denotar el conjunto de los numeros naturales y ω para denotar al conjunto $\mathbf{N} \cup \{0\}$.

Dado un conjunto A, usaremos $\mathcal{P}(A)$ para denotar el conjunto formado por todos los subconjuntos de A, es decir:

$$\mathcal{P}(A) = \{S : S \subseteq A\}$$

Si A es un conjunto finito, entonces $|A|$ denotara la cantidad de elementos de A.

Para $x, y \in \omega$, definamos

$$x \dot{-} y = \begin{cases} x - y & \text{si } x \geq y \\ 0 & \text{caso contrario} \end{cases}$$

Dados $x, y \in \omega$ diremos que x *divide a* y cuando haya un $z \in \omega$ tal que $y = z.x$. Notar que 0 divide a 0, 3 divide a 0 y 0 no divide a 23. Escribiremos $x \mid y$ para expresar que x divide a y. Dados $x, y \in \omega$, diremos que x e y son *coprimos* cuando 1 sea el unico elemento de ω que divide a ambos. Notese que x e y no son son coprimos sii existe un numero primo $p \in \omega$ que los divide a ambos

Si bien no hay una definicion natural en matematica de cuanto vale 0^0 (0 elevado a la 0), por convencion para nosotros $0^0 = 1$

1.1 Conjuntos

Supondremos que el lector sabe las nociones basicas sobre conjuntos, aunque resaltaremos algunas de las mas importantes para que el lector las repase.

La propiedad de *extensionalidad* nos dice que, dados conjuntos A, B, se tiene que $A = B$ si y solo si para cada objeto x se da que

$$x \in A \text{ si y solo si } x \in B$$

Esta propiedad es importante metodologicamente ya que a la hora de probar que dos conjuntos A, B son iguales, extensionalidad nos asegura que basta con ver que se dan las dos inclusiones $A \subseteq B$ y $B \subseteq A$.

Otro tema importante es manejar correctamente la notacion cuando definimos un conjunto usando llaves y mediante propiedades que caracterizan la pertenencia al mismo. Algunos ejemplos

- $\{x \in \mathbf{N} : x = 1 \text{ o } x \geq 5\}$

- $\{x : x \in \mathbf{R} \text{ y } x^2 \geq 100\}$

- $\{x : x = 100\}$

- $\{x^2 + 1 : x \in \omega\}$

- $\{x + y + z : x, y, z \in \{1, 2\}\}$

Dejamos al lector la tarea de entender en forma precisa que conjunto se esta denotando en cada caso.

1.2 Producto carteciano

Dados conjuntos $A_1, ..., A_n$, con $n \geq 2$, usaremos $A_1 \times ... \times A_n$ para denotar el *producto Cartesiano* de $A_1, ..., A_n$, es decir el conjunto formado por todas las n-uplas $(a_1, ..., a_n)$ tales que $a_1 \in A_1, ..., a_n \in A_n$. Si $A_1 = ... = A_n = A$, con $n \geq 2$, entonces escribiremos A^n en lugar de $A_1 \times ... \times A_n$. Para $n = 1$, definimos $A^n = A$, es decir $A^1 = A$. Usaremos $\Diamond$ para denotar la unica 0-upla. Definimos entonces $A^0 = \{\Diamond\}$. Si A es un conjunto denotaremos con $A^{\mathbf{N}}$ al conjunto formado por todas las infinituplas $(a_1, a_2, ...)$ tales que $a_i \in A$ para cada $i \in \mathbf{N}$. Por ejemplo

$$(1, 2, 3, 4, ...) \in \omega^{\mathbf{N}}$$

donde $(1, 2, 3, 4, ...)$ es una forma intuitiva de denotar la infinitupla cuyo i-esimo elemento es el numero natural i.

Si $(A_1, A_2, ...)$ es una infinitupla de conjuntos, entonces usaremos $\bigcup_{i=1}^{\infty} A_i$ o $\bigcup_{i \geq 1} A_i$ para denotar al conjunto

$$\{a : a \in A_i, \text{ para algun } i \in \mathbf{N}\}$$

1.3 Alfabetos

Un *alfabeto* es un conjunto finito de simbolos. Notese que $\emptyset$ es un alfabeto. Si Σ es un alfabeto, entonces Σ^* denotara el conjunto de todas las palabras formadas con simbolos de Σ. La unica palabra de longitud 0 es denotada con ε. Ya que en ε no ocurren simbolos, tenemos que $\varepsilon \in \Sigma^*$, para cualquier alfabeto, mas aun notese que $\emptyset^* = \{\varepsilon\}$. Usaremos $|\alpha|$ para denotar la longitud de la palabra α. Usaremos Σ^+ para denotar al conjunto $\Sigma^* - \{\varepsilon\}$. Notese que funciones, n-uplas y palabras son objetos de distinto tipo, por lo cual $\emptyset$, $\Diamond$ y ε son tres objetos matematicos diferentes.

Si $\alpha_1, ..., \alpha_n \in \Sigma^*$, con $n \geq 0$, usaremos $\alpha_1 ... \alpha_n$ para denotar la *concatenacion* de las palabras $\alpha_1, ..., \alpha_n$ (notese que cuando $n = 0$, resulta que $\alpha_1 ... \alpha_n = \varepsilon$). Si $\alpha_1 = ... = \alpha_n = \alpha$, entonces escribiremos α^n en lugar de $\alpha_1 ... \alpha_n$. O sea que $\alpha^0 = \varepsilon$.

Diremos que α *es subpalabra (propia) de* β cuando ($\alpha \notin \{\varepsilon, \beta\}$ y) existan palabras δ, γ tales que $\beta = \delta\alpha\gamma$. Diremos que β es un *tramo inicial (propio)* de α si hay una palabra γ tal que $\alpha = \beta\gamma$ (y $\beta \notin \{\varepsilon, \alpha\}$). En forma similar se define *tramo final (propio).*

Dados $i \in \omega$ y $\alpha \in \Sigma^*$ definamos

$$[\alpha]_i = \begin{cases} i\text{-esimo elemento de } \alpha & \text{si } 1 \leq i \leq |\alpha| \\ \varepsilon & \text{caso contrario} \end{cases}$$

Dada $\gamma \in \Sigma^*$, definamos

$$\gamma^R = \begin{cases} [\gamma]_{|\gamma|}[\gamma]_{|\gamma|-1}...[\gamma]_1 & \text{si } |\gamma| \geq 1 \\ \varepsilon & \text{caso contrario} \end{cases}$$

La palabra γ^R es llamada la *resiproca* de γ.

1.3.1 Ocurrencias

Dadas palabras $\alpha, \beta \in \Sigma^*$, con $|\alpha|, |\beta| \geq 1$, y un natural $i \in \{1, ..., |\beta|\}$, se dice que α *ocurre a partir de* i *en* β cuando se de que existan palabras δ, γ tales que $\beta = \delta\alpha\gamma$ y $|\delta| = i - 1$. Intuitivamente hablando α ocurre a partir de i en β cuando se de que si comensamos a leer desde el lugar i-esimo de β en adelante, leeremos la palabra α completa y luego posiblemente seguiran otros simbolos.

Notese que una palabra α puede ocurrir en β, a partir de i, y tambien a partir de j, con $i \neq j$. En virtud de esto, hablaremos de las distintas ocurrencias de α en β. Por ejemplo hay dos ocurrencias de la palabra *aba* en la palabra

$$ccccccabaccccabaccccc$$

y tambien hay dos ocurrencias de la palabra *aba* en la palabra

$$ccccccababacccccccccc$$

En el primer caso diremos que dichas ocurrencias de *aba* son *disjuntas*, en cambio en el segundo caso puede apreciarse que las dos ocurrencias se superponen en una posicion. No definiremos en forma matematica precisa el concepto de ocurrencia pero el lector no tendra problemas en comprenderlo y manejarlo en forma correcta.

A veces diremos que una ocurrencia esta *contenida* o *sucede* dentro de otra. Por ejemplo la segunda ocurrencia de *ab* en $babbbfabcccfabccc$ esta contenida en la primer ocurrencia de $fabc$ en $babbbfabcccfabccc$. Tambien haremos *reemplazos* de ocurrencias por palabras. Por ejemplo el resultado de reemplazar la primer ocurrencia de *abb* en $ccabbgfgabbgg$ por *oolala* es la palabra $ccoolalagfgabbgg$. El resultado de reemplazar todas las ocurrencias de *aba* en $ccabagfgabaggaba$ por \$\$ es la palabra $cc\$\$gfg\$\$gg\$\$$. En algunos casos deberemos especificar que los reemplazos se hagan *simultaneamente*. Por ejemplo hablaremos del resultado de reemplazar en γ, simultaneamente, todas las ocurrencias de α_1 por β_1 y todas las de α_2 por β_2. Aqui la aclaracion de simultaneidad es importante ya que si primero reemplazaramos las ocurrencias de α_1

por β_1 y despues las de α_2 por β_2, el resultado puede cambiar porque en β_1 puede haber ocurrencias de α_2. Dejamos al lector dar un ejemplo en el cual el reemplazo secuencial y el simultaneo dan distintos resultados.

Si $\alpha \in \Sigma^*$ y $\sigma \in \Sigma$, usaremos $|\alpha|_\sigma$ para denotar la cantidad de ocurrencias del simbolo σ en α.

1.4 Matematica orientada a objetos

Nuestro estilo o enfoque matematico pondra enfasis en los objetos, es decir haremos matematica prestando atencion a los distintos objetos matematicos involucrados, los cuales siempre seran definidos en forma precisa en terminos de objetos mas primitivos. Hay ciertos objetos matematicos los cuales no definiremos y supondremos que el lector tiene una idea clara y precisa de los mismos. Por ejemplo un tipo de objeto matematico, quizas el mas famoso, son los *numeros*. No diremos que es un numero pero supondremos que el lector tiene una intuicion clara acerca de este tipo de objetos y de sus propiedades basicas. Otro tipo de objeto que no definiremos y que sera clave para nuestro enfoque son los *conjuntos*. Nuevamente, no diremos que es un conjunto pero supondremos que el lector tiene una intuicion clara acerca de estos objetos y sus propiedades basicas. Es importante que en nuestro enfoque, numeros y conjuntos son objetos de distinta naturaleza por lo cual nunca un numero es un conjunto ni un conjunto es un numero. En particular esto nos dice que el numero 0 y el conjunto $\emptyset$ son objetos distintos. Otro tipo de objeto matematico muy importante para la matematica discreta son los *simbolos*. No discutiremos que es un simbolo sino que aceptaremos este concepto en forma primitiva. Tambien constituyen un tipo de objeto matematico las *palabras*, las cuales intuitivamente hablando son juxtaposiciones de simbolos. Otro tipo de objeto matematico muy importante son los *pares ordenados* o 2-*uplas*, es decir los objetos de la forma (a, b), donde a y b son objetos matematicos cualesquiera. Tambien son objetos matematicos y de distinta naturaleza las 3-*uplas*, las 4-*uplas* y en general las n-*uplas* para n un numero natural mayor o igual a 2. Cabe destacar que en nuestro enfoque no habra 1-uplas. Sin envargo, si bien hay una sola 0-*upla*, ella constituye un tipo de objeto matematico distinto a los antes mensionados. El ultimo tipo de objeto matematico que consideraremos es aquel de las *infinituplas*.

Tenemos entonces dividido nuestro universo matematico en las distintas categorias de objetos:

$$\begin{array}{l}
\text{NUMERO}\\
\text{CONJUNTO}\\
\text{PALABRA}\\
0-\text{UPLA}\\
2-\text{UPLA}\\
3-\text{UPLA}\\
\quad\vdots\\
\text{INFINITUPLA}
\end{array}$$

(Notar que los simbolos quedan contenidos en la categoria de las palabras). Es importante entender que las anteriores categorias o tipos de objetos son disjuntas entre si, es decir nunca un numero sera una palabra o una palabra sera una 3-upla etc. Esto nos permite definir una funcion Ti la cual a un objeto matematico le asigna su tipo de objeto matematico segun la lista anterior. Por ejemplo:

$$\begin{array}{rcl}
Ti(\pi) & = & \text{NUMERO}\\
Ti(\mathbf{N}) & = & \text{CONJUNTO}\\
Ti(\mathcal{P}(\mathbf{N})) & = & \text{CONJUNTO}\\
Ti((1,2,3)) & = & 3-\text{UPLA}\\
Ti(\emptyset) & = & \text{CONJUNTO}\\
Ti(\varepsilon) & = & \text{PALABRA}\\
Ti(\Diamond) & = & 0-\text{UPLA}\\
Ti(\alpha) & = & \text{PALABRA, si } \alpha \text{ es un simbolo}\\
Ti(f) & = & \text{CONJUNTO, si } f \text{ es una funcion}
\end{array}$$

1.5 El concepto de funcion

Asumiremos que el lector tiene una idea intuitiva del concepto de funcion. Daremos aqui una definicion matematica de dicho concepto. Una *funcion* es un conjunto f de pares ordenados con la siguiente propiedad

(F) Si $(x, y) \in f$ y $(x, z) \in f$, entonces $y = z$.

Por ejemplo, si tomamos $f = \{(x, x^2) : x \in \omega\}$ se puede ver facilmente que f cumple la propiedad (F). Dada una funcion f, definamos

$$\begin{array}{rcl}
D_f & = & \text{dominio de } f = \{x : (x,y) \in f \text{ para algun } y\}\\
I_f & = & \text{imagen de } f = \{y : (x,y) \in f \text{ para algun } x\}
\end{array}$$

A veces escribiremos $\text{Dom}(f)$ y $\text{Im}(f)$ para denotar, respectivamente, el dominio y la imagen de una funcion f. Como es usual dado $x \in D_f$, usaremos $f(x)$ para denotar al unico $y \in I_f$ tal que $(x, y) \in f$. Notese que $\emptyset$ es una funcion y que $D_\emptyset = I_\emptyset = \emptyset$. Por ejemplo para $f = \{(x, x^2) : x \in \omega\}$ se tiene que $D_f = \omega$ y $I_f = \{y : y = x^2 \text{ para algun } x \in \omega\}$. Ademas notese que $f(x) = x^2$, para cada $x \in D_f$.

Escribiremos $f : S \subseteq A \to B$ para expresar que f es una funcion tal que $D_f = S \subseteq A$ y $I_f \subseteq B$. Tambien escribiremos $f : A \to B$ para expresar que f es una funcion tal que $D_f = A$ y $I_f \subseteq B$. En tal contexto llamaremos a B *conjunto de llegada*. Por supuesto B no esta determinado por f ya que solo debe cumplir $I_f \subseteq B$.

Muchas veces para definir una funcion f, lo haremos dando su dominio y su regla de asignacion, es decir especificaremos en forma precisa que conjunto es el dominio de f y ademas especificaremos en forma presisa quien es $f(x)$ para cada x de dicho dominio. Obviamente esto determina por completo a la funcion f ya que $f = \{(x, f(x)) : x \in D_f\}$. Por ejemplo si decimos que f es la funcion dada por:

$$\begin{array}{rcl} D_f & = & \omega \\ f(x) & = & 23x^2 \end{array}$$

nos estaremos refiriendo a la funcion $\{(x, 23x^2) : x \in \omega\}$. Tambien escribiremos

$$\begin{array}{rcl} f : \omega & \to & \omega \\ x & \to & 23x^2 \end{array}$$

para describir a f. Es decir, a veces para hacer mas intuitiva aun la descripcion de la funcion, tambien incluiremos un conjunto de llegada de dicha funcion y a la regla de asignacion la escribiremos usando una flecha. Para dar otro ejemplo, si escribimos sea f dada por:

$$\begin{array}{rcl} f : \mathbf{N} & \to & \omega \\ x & \to & \left\{ \begin{array}{ll} x+1 & \text{si } x \text{ es par} \\ x^2 & \text{si } x \text{ es impar} \end{array} \right. \end{array}$$

estaremos diciendo que f es la funcion

$$\{(x, x+1) : x \text{ es par y } x \in \mathbf{N}\} \cup \{(x, x^2) : x \text{ es impar y } x \in \mathbf{N}\}$$

Igualdad de funciones

Sean f y g dos funciones. Ya que las mismas son conjuntos, tendremos que f sera igual a g si y solo si para cada par (a, b), se tiene que $(a, b) \in f$ sii $(a, b) \in g$. Muchas veces sera util el siguiente criterio de igualdad de funciones:

Lemma 1 *Sean f y g funciones. Entonces $f = g$ sii $D_f = D_g$ y para cada $x \in D_f$ se tiene que $f(x) = g(x)$*

1.5.1 Funcion identidad

Dado un conjunto A, a la funcion

$$\begin{array}{ccc} A & \to & A \\ a & \to & a \end{array}$$

La denotaremos con Id_A y la llamaremos la funcion *identidad sobre* A. Notese que $Id_A = \{(a,a) : a \in A\}$.

1.5.2 Composicion de funciones

Dadas funciones f y g definamos la funcion $f \circ g$ de la siguiente manera:

$$\begin{array}{rcl} D_{f \circ g} & = & \{e \in D_g : g(e) \in D_f\} \\ f \circ g(e) & = & f(g(e)) \end{array}$$

Notar que $f \circ g = \{(u,v) : \text{existe } z \text{ tal que } (u,z) \in g \text{ y } (z,v) \in f\}$. Notese que $f \circ g \neq \emptyset$ si y solo si $I_g \cap D_f \neq \emptyset$, lo cual nos dice que muchas veces sucedera que $f \circ g = \emptyset$

1.5.3 Funciones inyectivas, suryectivas y biyectivas

Una funcion f es *inyectiva* cuando no se da que $f(a) = f(b)$ para algun par de lementos distintos $a, b \in D_f$. Dada una funcion $f : A \to B$ diremos que f es *suryectiva* cuando $I_f = B$. Debe notarse que el concepto de suryectividad depende de un conjunto de llegada previamente fijado, es decir que no tiene sentido hablar de la suryectividad de una funcion f si no decimos respecto de que conjunto de llegada lo es. Muchas veces diremos que una funcion f es *sobre* para expresar que es suryectiva.

Dada una funcion $f : A \to B$ diremos que f es *biyectiva* cuando f sea inyectiva y suryectiva. Notese que si $f : A \to B$ es biyectiva, entonces podemos definir una nueva funcion $f^{-1} : B \to A$, de la siguiente manera:

$$f^{-1}(b) = \text{ unico } a \in A \text{ tal que } f(a) = b$$

La funcion f^{-1} sera llamada la *inversa de* f. Notese que $f \circ f^{-1} = Id_B$ y $f^{-1} \circ f = Id_A$. El siguiente lema muestra que esta ultima propiedad caracteriza la inversa.

Lemma 2 *Supongamos $f : A \to B$ y $g : B \to A$ son tales que $f \circ g = Id_B$ y $g \circ f = Id_A$. Entonces f y g son biyectivas, $f^{-1} = g$ y $g^{-1} = f$.*

1.5.4 El nucleo de una funcion

Dada una funcion $f : A \to B$, definamos:

$$\ker(f) = \{(a,b) \in A^2 : f(a) = f(b)\}$$

El conjunto $\ker(f)$ sera llamado el *nucleo* de f. Notese que f es inyectiva si y solo si $\ker(f) = \{(a,a) : a \in A\}$.

1.5.5 Funcion caracteristica de un subconjunto

Sea X un conjunto cualquiera y sea $S \subseteq X$. Usaremos χ_S^X para denotar la funcion

$$\begin{array}{rcl} \chi_S^X : X & \to & \omega \\ x & \to & \left\{ \begin{array}{l} 1 \text{ si } x \in S \\ 0 \text{ si } x \notin S \end{array} \right. \end{array}$$

Llamaremos a χ_S^X la *funcion caracteristica de S con respecto a X*. Muchas veces cuando el conjunto X este fijo y sea claro el contexto, escribiremos χ_S en lugar de χ_S^X.

1.5.6 Restriccion de una funcion

Dada una funcion f y un conjunto $S \subseteq D_f$, usaremos $f|_S$ para denotar la *restriccion* de f al conjunto S, i.e. $f|_S = f \cap (S \times I_f)$. Notese que $f|_S$ es la funcion dada por

$$\begin{array}{rcl} D_{f|_S} & = & S \\ f|_S(e) & = & f(e), \text{ para cada } e \in S \end{array}$$

Notese que cualesquiera sea la funcion f tenemos que $f|_\emptyset = \emptyset$ y $f|_{D_f} = f$.

1.5.7 Funciones de la forma $[f_1, ..., f_n]$

Dadas funciones $f_1, ..., f_n$, con $n \geq 2$, definamos la funcion $[f_1, ..., f_n]$ de la siguiente manera:

$$\begin{array}{rcl} D_{[f_1,...,f_n]} & = & D_{f_1} \cap ... \cap D_{f_n} \\ [f_1, ..., f_n](e) & = & (f_1(e), ..., f_n(e)) \end{array}$$

Notese que $I_{[f_1,...,f_n]} \subseteq I_{f_1} \times \cdots \times I_{f_n}$. Por conveniencia notacional (que el lector entendera mas adelante) definiremos $[f_1] = f_1$. Es decir que hemos definido para cada sucecion de funciones $f_1, ..., f_n$, con $n \geq 1$, una nueva funcion la cual denotamos con $[f_1, ..., f_n]$.

1.5.8 Union de funciones con dominios disjuntos

Una observacion interesante es que si $f_i : A_i \to B_i$, $i = 1, ..., k$, son funciones tales que $A_i \cap A_j = \emptyset$ para $i \neq j$, entonces $f_1 \cup ... \cup f_k$ es la funcion

$$\begin{array}{rcl} A_1 \cup ... \cup A_k & \to & B_1 \cup ... \cup B_k \\ e & \to & \begin{cases} f_1(e) & \text{si } e \in A_1 \\ \vdots & \vdots \\ f_k(e) & \text{si } e \in A_k \end{cases} \end{array}$$

1.6 Relaciones binarias

Sea A un conjunto. Por una *relacion binaria sobre* A entenderemos un subconjunto de A^2. Algunos ejemplos:

(E1) Sea $R = \{(1, 2), (2, 3)\}$. Entonces R es una relacion binaria sobre $\mathbf{N}$.

(E2) Sea $R = \{(x, y) \in \omega^2 : x \text{ divide a } y\}$. Entonces R es una relacion binaria sobre ω.

(E3) Sea $R = \{(r, t) \in \mathbf{R}^2 : r \leq t\}$. Entonces R es una relacion binaria sobre $\mathbf{R}$

(E4) $\emptyset$ es una relacion binaria sobre A, cualesquiera sea el conjunto A.

(E5) Sea $R = \{(x, y) \in \omega^2 : x < y \text{ o } y = 0\}$. Entonces R es una relacion binaria sobre ω

Notese que si R es una relacion binaria sobre A y $A \subseteq B$ entonces R es una relacion binaria sobre B. Por ejemplo las relaciones dadas en los ejemplos (E1), (E2), (E4) y (E5) tambien son relaciones binarias sobre $\mathbf{R}$. Sin envargo si R es una relacion binaria sobre B y $A \subseteq B$ entonces no necesariamente R sera una relacion binaria sobre A (por que?).

Como es usual, cuando R sea una relacion binaria sobre un conjunto A, algunas veces escribiremos aRb en lugar de $(a, b) \in R$.

1.6.1 Propiedades notables de relaciones binarias

Hay algunas propiedades que pueden tener o no las relaciones binarias sobre un conjunto A, las cuales son muy importantes en matematica. Algunas de estas son:

Reflexividad xRx, cualesquiera sea $x \in A$

Transitividad xRy y yRz implica xRz, cualesquiera sean $x, y, z \in A$

Simetria xRy implica yRx, cualesquiera sean $x, y \in A$

Antisimetria xRy y yRx implica $x = y$, cualesquiera sean $x, y \in A$

Cuando R cumpla la primer propiedad diremos que R es *reflexiva, con respecto a A*. Analogamente diremos que R es *transitiva, simetrica o antisimetrica, con respecto a A*, cuando se den, respectivamente las otras propiedades. Notese que estas propiedades dependen del conjunto A, por ejemplo si tomamos $R = \{(r,t) \in \mathbf{N}^2 : r \leq t\}$ entonces R es una relacion binaria sobre $\mathbf{N}$ y tambien es una relacion binaria sobre ω, pero es relexiva con respepcto a $\mathbf{N}$ y no lo es con respecto a ω ya que $(0,0)$ no pertenece a R. Sin envargo R es transitiva con respecto a $\mathbf{N}$ y tambien lo es con respecto a ω.

1.6.2 Ordenes parciales

Una relacion binaria R sobre un conjunto A sera llamada un *orden parcial sobre A* si es reflexiva, transitiva y antisimetrica respecto de A. Algunos ejemplos:

(E1) Sea $R = \{(r,t) \in \mathbf{R}^2 : r \leq t\}$. Entonces R es un orden parcial sobre $\mathbf{R}$, llamado el orden usual de $\mathbf{R}$

(E2) Sea $R = \{(1,2),(1,3),(1,1),(2,2),(3,3)\}$. Entonces R es un orden parcial sobre $\{1,2,3\}$

(E3) Sea $R = \{(S,T) \in \mathcal{P}(\omega)^2 : S \subseteq T\}$. Entonces R es un orden parcial sobre $\mathcal{P}(\omega)$

(E4) Sea $R = \{(x,y) \in \omega^2 : x \leq y\}$. Entonces R es un orden parcial sobre ω.

(E5) Sea $R = \{(1,1)\}$. Entonces R es un orden parcial sobre $\{1\}$.

(E6) $\{(a,b) : a = b\}$ es un orden parcial sobre A, cualesquira sea el conjunto A

(E7) Sea $\leq\ = \{(n,m) \in \mathbf{N}^2 : n \mid m\}$. Es facil ver que $\leq$ es un orden parcial sobre $\mathbf{N}$

Notese que las relaciones dadas en (E1) y (E4) son distintas, ademas la relacion dada en (E4) no es un orden parcial sobre $\mathbf{R}$ (por que?).

Muchas veces denotaremos con $\leq$ a una relacion binaria que sea un orden parcial. Esto hace mas intuitiva nuestra escritura pero siempre hay que tener en cuenta que $\leq$ en estos casos esta denotando cierto conjunto de pares ordenados previamente definido.

Usaremos la siguiente

Convencion notacional Si hemos denotado con $\leq$ a cierto orden parcial sobre un conjunto A, entonces

(a) Denotaremos con $<$ a la relacion binaria $\{(a,b) \in A^2 : a \leq b \text{ y } a \neq b\}$. Es decir que $< \ = \{(a,b) \in A^2 : a \leq b \text{ y } a \neq b\}$. Cuando se de $a < b$ diremos que *a es menor que b* o que *b es mayor que a* (*respecto de* $\leq$)

(b) Denotaremos con $\prec$ a la relacion binaria

$$\{(a,b) \in A^2 : a < b \text{ y no existe } z \text{ tal que } a < z < b\}$$

Cuando se de $a \prec b$ diremos que a *es cubierto por* b o que b *cubre a* a (*respecto de* $\leq$)

Algunos ejemplos:

(E1) Si $A = \mathbf{R}$ y $\leq = \{(r,t) \in \mathbf{R}^2 : r = t\}$, entonces $< = \emptyset$

(E2) Si $A = \{1,2,3,4\}$ y $\leq = \{(1,2),(2,3),(1,3),(1,1),(2,2),(3,3),(4,4)\}$, entonces $< = \{(1,2),(2,3),(1,3)\}$ y $\prec = \{(1,2),(2,3)\}$. En particular tenemos que $1 \prec 2$, $1 < 3$ pero no se da que $1 \prec 3$.

(E3) Si $A = \mathcal{P}(\omega)$ y $\leq = \{(S,T) \in \mathcal{P}(\omega)^2 : S \subseteq T\}$, entonces $< = \{(S,T) \in \mathcal{P}(\omega)^2 : S \subsetneq T\}$ y $S \prec T$ sii hay un $n \in T - S$ tal que $T = S \cup \{n\}$

Ordenes totales sobre un conjunto Sea A un conjunto cualquiera. Por un *orden total sobre* A entenderemos un orden parcial $\leq$ sobre A el cual cumpla:

(C) $a \leq b$ o $b \leq a$, cualesquiera sean $a, b \in A$

Supongamos A es finito no vacio y $\leq$ es un orden total sobre A. La propiedad (C) nos permite probar que para cada conjunto no vacio $S \subseteq A$, hay un elemento $s \in S$ el cual cumple $s \leq s'$ para cada $s' \in S$. Por supuesto, s es unico (por que?) y habitualmente es llamado el *menor elemento de* S, ya que es menor que todo otro elemento de S.

Si A es finito no vacio y $\leq$ es un orden total sobre A, podemos definir recursivamente una funcion $f : \{1, ..., |A|\} \rightarrow A$ de la siguiente manera:

- $f(1) =$ menor elemento de A
- Si $i \in \{1, ..., |A| - 1\}$, entonces
  - $f(i+1) =$ menor elemento de $A - \{f(1), ..., f(i)\}$

Como es habitual, $f(i)$ es llamado el *i-esimo elemento de* A.

Muchas veces para dar un orden total sobre un conjunto finito A, daremos simplemente sus elementos en forma creciente ya que esto determina el orden por completo. Por ejemplo si $A = \{1,2,3\}$, el orden total dado por $2 < 1 < 3$ es la relacion $\leq = \{(2,1),(1,3),(2,3),(1,1),(2,2),(3,3)\}$.

Un concepto importante relativo a los ordenes totales es el de *sucesor*. Si $\leq$ es un orden total sobre A y $a, b \in A$, diremos que b *es el sucesor de* a cuando se de que $a < b$ y $b \leq c$, para cada $c \in A$ tal que $a < c$, i.e., b es el menor elemento del conjunto $\{c \in A : \text{tal que } a < c\}$. No siempre existe el sucesor de un elemento. Por ejemplo si $\leq$ es el orden usual de $\mathbf{R}$, entonces ningun elemento tiene sucesor (justifique).

Diagramas de Hasse Dado un orden parcial $\leq$ sobre un conjunto finito A podemos realizar un diagrama de $\leq$, llamado *diagrama de Hasse,* siguiendo las siguientes instrucciones:

(1) Asociar en forma inyectiva, a cada $a \in A$ un punto p_a del plano

(2) Trazar un segmento de recta uniendo los puntos p_a y p_b, cada vez que $a \prec b$

(3) Realizar lo indicado en los puntos (1) y (2) en tal forma que

  (i) Si $a \prec b$, entonces p_a esta por debajo de p_b

  (ii) Si un punto p_a ocurre en un segmento del diagrama entonces lo hace en alguno de sus extremos.

La relacion de orden $\leq$ puede ser facilmente obtenida de su diagrama, a saber, $a \leq b$ sucedera si y solo si $p_a = p_b$ o hay una sucesion de segmentos ascendentes desde p_a hasta p_b.

Ejemplos:

1.6.3 Relaciones de equivalencia

Sea A un conjunto cualquiera. Por una *relacion de equivalencia sobre A* entenderemos una relacion binaria sobre A la cual es reflexiva, transitiva y simetrica, con respecto a A, es decir, la cual cumple:

Reflexividad xRx, cualesquiera sea $x \in A$

Transitividad xRy y yRz implica xRz, cualesquiera sean $x, y, z \in A$

Simetria xRy implica yRx, cualesquiera sean $x, y \in A$

Algunos ejemplos:

(E1) Sea $R = \{(r,t) \in \mathbf{R}^2 : r = t\}$. Entonces R es una relacion de equivalencia sobre $\mathbf{R}$

(E2) Dada una funcion $f : A \to B$, el nucleo de f, i.e. $\ker(f) = \{(a,b) \in A^2 : f(a) = f(b)\}$ es una relacion de equivalencia sobre A.

(E3) Sea $R = \{(1,1),(2,2),(3,3),(1,2),(2,1)\}$. Entonces R es una relacion de equivalencia sobre $\{1,2,3\}$

(E4) Sea $R = \{(x,y) \in \omega^2 : x = y\}$. Entonces R es una relacion de equivalencia sobre ω

(E5) Sea $R = \{(S,T) \in \mathcal{P}(\omega)^2 : (S - T) \cup (T - S) \text{ es finito}\}$. Entonces R es una relacion de equivalencia sobre $\mathcal{P}(\omega)$

(E7) Sea $R = \{(1,1)\}$. Entonces R es una relacion de equivalencia sobre $\{1\}$.

(E8) Sea $R = \{(x,y) \in \mathbf{Z}^2 : x - y \text{ es multiplo de } 2\}$. Entonces R es una relacion de equivalencia sobre $\mathbf{Z}$.

Dada una relacion de equivalencia R sobre A y $a \in A$, definimos:

$$a/R = \{b \in A : aRb\}$$

El conjunto a/R sera llamado la *clase de equivalencia de a, con respecto a R*. Ejemplos:

(E1) Si $R = \{(r,t) \in \mathbf{R}^2 : r = t\}$, entonces $r/R = \{r\}$, cualesquier sea $r \in \mathbf{R}$

(E2) Si $R = \{(1,1),(2,2),(3,3),(1,2),(2,1)\}$, entonces $1/R = 2/R = \{1,2\}$ y $3/R = \{3\}$

(E3) Si $R = \{(x,y) \in \mathbf{Z}^2 : x - y \text{ es multiplo de } 2\}$, entonces $0/R = \{t \in \mathbf{Z} : t \text{ es par}\}$, $1/R = \{t \in \mathbf{Z} : t \text{ es impar}\}$ y en general notese que $n/R = \{t \in \mathbf{Z} : t \text{ es par}\}$ si n es par y $n/R = \{t \in \mathbf{Z} : t \text{ es impar}\}$ si n es impar. Es decir que hay solo dos clases de equivalencia con respecto a R

Algunas propiedades basicas son:

Lemma 3 *Sea R una relacion de equivalencia sobre A. Sean $a, b \in A$.*

(1) $a \in a/R$

(2) aRb si y solo si $a/R = b/R$. Es decir que $b \in a/R$ implica $b/R = a/R$

(3) $a/R \cap b/R = \emptyset$ o $a/R = b/R$

Proof. (1) es muy facil.

(2). Supongamos aRb. Veremos que $a/R \subseteq b/R$. Supongamos $c \in a/R$. Entonces aRc. Como aRb, tenemos que bRa, por lo cual hemos probado que bRa y aRc, lo cual implica que bRc. O sea que cRb, lo cual nos dice que $c \in b/R$. Esto prueba que $a/R \subseteq b/R$. Similarmente se prueba que $b/R \subseteq a/R$, con lo cual se tiene que $a/R = b/R$.

Reciprocamente, si $a/R = b/R$, entonces $b \in a/R$ ya que $b \in b/R$. Pero esto nos dice que aRb.

(3). Supongamos que $a/R \cap b/R$ no es vacio, es decir hay un $c \in a/R \cap b/R$. Entonces es facil ver que aRb. Pero entonces por (2) tenemos que $a/R = b/R$. ■

Denotaremos con A/R al conjunto $\{a/R : a \in A\}$. Llamaremos a A/R el *cociente de A por R*. Ejemplos:

(E1) Si $R = \{(r,t) \in \mathbf{R}^2 : r = t\}$, entonces $\mathbf{R}/R = \{\{r\} : r \in \mathbf{R}\}$

(E2) Si $R = \{(1,1),(2,2),(3,3),(1,2),(2,1)\}$, entonces $\{1,2,3\}/R = \{\{1,2\},\{3\}\}$

(E3) Si $R = \{(x, y) \in \mathbf{Z}^2 : x - y$ es multiplo de $2\}$, ya vimos que $\mathbf{Z}/R = \{\{t \in \mathbf{Z} : t$ es par$\}, \{t \in \mathbf{Z} : t$ es impar$\}\}$

Si R es una relacion de equivalencia sobre A, definamos la funcion $\pi_R : A \to A/R$ por $\pi_R(a) = a/R$, para cada $a \in A$. La funcion π_R es llamada la *proyeccion canonica* (*respecto de* R).

Lemma 4 *Sea R una relacion de equivalencia sobre A. Entonces* $\ker \pi_R = R$. *Es decir que π_R es inyectiva sii $R = \{(x, y) \in A^2 : x = y\}$*

Definicion de funciones con dominio A/R Supongamos R es una relacion de equivalencia sobre $\mathbf{R}$ y supongamos definimos una funcion $f : \mathbf{R}/R \to \mathbf{R}$ de la siguiente manera:

$$f(r/R) = r^2$$

A priori puede pareser que esta definicion es natural y que no esconde ninguna posible complicacion. Pero supongamos que R es tal que $2R6$. Entonces tendriamos que $2/R = 6/R$ lo cual produciria la siguiente contradiccion:

$$\begin{aligned} 4 &= 2^2 \\ &= f(2/R) \\ &= f(6/R) \\ &= 6^2 \\ &= 36 \end{aligned}$$

El problema aqui es que la ecuacion $f(r/R) = r^2$ no esta definiendo en forma correcta o inhambigua una funcion ya que el supuesto valor de la funcion en una clase de equivalencia dada depende de que representante de la clase usamos para denotarla. Si usamos el 2 la ecuacion nos dice que entonces f debe valer 4 y si usamos el 6 la ecuacion nos dice que f debe valer 36. Claramente no estamos definiendo una funcion.

Para dar un ejemplo mas concreto de este fenomeno de ambiguedad, supongamos

$$R = \{(x, y) \in \mathbf{Z}^2 : x - y \text{ es multiplo de } 2\}$$

y definimos una funcion $f : \mathbf{Z}/R \to \mathbf{R}$ de la siguiente manera:

$$f(n/R) = 1/(n^2 + 1)$$

Como ya vimos $\mathbf{Z}/R = \{\{t \in \mathbf{Z} : t$ es par$\}, \{t \in \mathbf{Z} : t$ es impar$\}\}$, por lo cual facilmente se puede llegar a que la ecuacion $f(n/R) = 1/(n^2 + 1)$ no define correctamente una funcion. Dejamos al lector explicar esto mas detalladamente.

Sin envargo hay muchos casos en los cuales este tipo de definiciones son inhambiguas y desde luego muy importantes en el algebra moderna. Como un primer ejemplo tenemos el siguiente lema el cual es una de las ideas fundamentales del algebra moderna.

Lemma 5 *Si $f : A \to B$ es sobre, entonces la ecuacion $\bar{f}(a/\ker f) = f(a)$ define en forma inhambigua una funcion $\bar{f} : A/\ker f \to B$ la cual es biyectiva.*

Proof. Que la ecuacion $\bar{f}(a/\ker f) = f(a)$ define sin ambiguedad una funcion $\bar{f} : A/\ker f \to B$ es obvio ya que si $a/\ker f = b/\ker f$, entonces por definicion de $\ker f$ debera suceder que $a = b$. Dejamos al lector la prueba de que $\bar{f}$ es biyectiva ■

Correspondencia entre relaciones de equivalencia y particiones Dado un conjunto A por una *particion de A* entenderemos un conjunto $\mathcal{P}$ tal que:

- Cada elemento de $\mathcal{P}$ es un subconjunto no vacio de A
- Si $S_1, S_2 \in \mathcal{P}$ y $S_1 \neq S_2$, entonces $S_1 \cap S_2 = \emptyset$
- $A = \{a : a \in S$, para algun $S \in \mathcal{P}\}$

La ultima condicion dice simplemente que la union de todos los elementos de $\mathcal{P}$ debe ser A. Ejemplos:

(E1) Si $A = \{1, 2, 3, 4, 5\}$, entonces

$$\mathcal{P} = \{\{1, 5\}, \{2, 3\}, \{4\}\}$$

es una particion de A

(E2) $\mathcal{P} = \{\mathbf{N}, \mathbf{R} - \mathbf{N}\}$ es una particion de $\mathbf{R}$

(E3) $\mathcal{P} = \{\{0\}, \{1, 2\}, \{3, 4\}, \{5, 6\}, \{7, 8\}, \{9, 10\}, ...\}$ es una particion de ω

Una observacion importante es que si $\mathcal{P}$ es una particion de A, entonces para cada $a \in A$ hay un unico $S \in \mathcal{P}$ tal que $a \in S$ (por que?). O sea que podemos hablar de EL elemento de $\mathcal{P}$ que contiene a a.

Dada una particion $\mathcal{P}$ de un conjunto A podemos definir una relacion binaria asociada a $\mathcal{P}$ de la siguiente manera:

$$R_{\mathcal{P}} = \{(a, b) \in A^2 : a, b \in S, \text{ para algun } S \in \mathcal{P}\}$$

Lemma 6 *Sea A un conjunto cualquiera. Entonces:*

(1) Sea $\mathcal{P}$ una particion de A. Entonces $R_{\mathcal{P}}$ es una relacion de equivalencia sobre A.

(2) Sea R una relacion de equivalencia sobre A. Entonces A/R es una particion de A.

Proof. (1). Es facil ver que $R_{\mathcal{P}}$ es reflexiva y simetrica. Veamos que es transitiva. Supongamos que $aR_{\mathcal{P}}b$ y $bR_{\mathcal{P}}c$. O sea que hay $S_1, S_2 \in \mathcal{P}$ tales que $a, b \in S_1$ y $b, c \in S_2$. Ya que S_1 y S_2 tienen un elemento en comun, debera suceder que $S_1 = S_2$. Pero entonces tenemos que $a, c \in S_1$, lo cual nos dice que $aR_{\mathcal{P}}c$.

(2). Sigue facilmente del Lema 3. ■

El siguiente teorema da una correspondencia natural entre relaciones de equivalencia sobre A y particiones de A.

Theorem 7 *Sea A un conjunto cualquiera. Sean*

$$\begin{array}{rcl} Part & = & \{particiones\ de\ A\} \\ ReEq & = & \{relaciones\ de\ equivalencia\ sobre\ A\} \end{array}$$

Entonces las funciones:

$$\begin{array}{rcl} Part & \to & ReEq \\ \mathcal{P} & \to & R_{\mathcal{P}} \end{array} \qquad\qquad \begin{array}{rcl} ReEq & \to & Part \\ R & \to & A/R \end{array}$$

son biyecciones una inversa de la otra.

Proof. Notese que por el Lema 2 basta con probar:

(1) $A/R_{\mathcal{P}} = \mathcal{P}$, cualesquiera sea la particion $\mathcal{P}$ de A

(2) $R_{A/R} = R$, cualesquiera sea la relacion de equivalencia R sobre A

Prueba de (1). Primero veamos que $A/R_{\mathcal{P}} \subseteq \mathcal{P}$. Sea $a \in A$, veremos que $a/R_{\mathcal{P}} = \{b : aR_{\mathcal{P}}b\} \in \mathcal{P}$. Sea S el unico elemento de $\mathcal{P}$ que contiene a a. Es facil ver de la definicion de $R_{\mathcal{P}}$ que $a/R_{\mathcal{P}} = S$ por lo cual $a/R_{\mathcal{P}} \in \mathcal{P}$. Veamos ahora que $\mathcal{P} \subseteq A/R_{\mathcal{P}}$. Sea $S \in \mathcal{P}$. Sea $a \in S$. Es facil ver de la definicion de $R_{\mathcal{P}}$ que $a/R_{\mathcal{P}} = S$ por lo cual $S \in A/R_{\mathcal{P}}$.

Prueba de (2). Primero veamos que $R_{A/R} \subseteq R$. Supongamos $aR_{A/R}b$. Entonces $a, b \in c/R$, para algun $c \in A$. Es claro que entonces aRb. Veamos ahora que $R \subseteq R_{A/R}$. Supongamos que aRb. Entonces $a, b \in a/R$, lo cual nos dice que $aR_{A/R}b$. ■

El teorema anterior muestra que a nivel de informacion es lo mismo tener una relacion de equivalencia sobre A que tener una particion de A. Esto es muy util ya que muchas veces es mas facil especificar una relacion de equivalencia via su particion asociada. Por ejemplo si hablamos de la relacion de equivalencia sobre $\{1, 2, 3, 4, 5\}$ dada por la particion

$$\mathcal{P} = \{\{1, 5\}, \{4\}, \{2, 3\}\}$$

nos estaremos refiriendo a $R_{\mathcal{P}}$, es decir a la relacion:

$$\{(1, 1), (2, 2), (3, 3), (4, 4), (5, 5), (1, 5), (5, 1), (2, 3), (3, 2)\}$$

1.7 Operaciones n-arias sobre un conjunto

Sea A un conjunto. Dado $n \in \omega$, por una *operacion n-aria sobre A* entenderemos una funcion cuyo dominio es A^n y cuya imagen esta contenida en A. A las operaciones 2-arias (resp. 3-arias, 4-arias) tambien las llamaremos *operacion binarias* (resp. *ternarias, cuaternarias*). Algunos ejemplos:

(E1) Sea $f : \mathbf{R} \times \mathbf{R} \rightarrow \mathbf{R}$ dada por $f(x, y) = x+y$. Entonces f es una operacion 2-aria sobre $\mathbf{R}$

(E2) Sea $f : \{\Diamond\} \rightarrow \omega$, dada por $f(\Diamond) = 5$. Entonces f es una operacion 0-aria sobre ω.

(E3) Sea $f : \mathbf{N} \times \mathbf{N} \times \mathbf{N} \times \mathbf{N} \times \mathbf{N} \rightarrow \mathbf{N}$, dada por $f(x_1, x_2, x_3, x_4, x_5) = (x_1.x_2)+ x_3$. Entonces f es una operacion 5-aria sobre $\mathbf{N}$.

1.8 Relaciones n-arias sobre un conjunto

Sea A un conjunto. Dado $n \in \omega$, por una *relacion n-aria sobre A* entenderemos un subconjunto de A^n. A las relaciones 2-arias (resp. 3-arias, 4-arias) tambien las llamaremos *relaciones binarias* (resp. *ternarias, cuaternarias*). Algunos ejemplos:

(E1) Sea $R = \{(r, t) \in \mathbf{R} \times \mathbf{R} : r \leq t\}$. Entonces R es una relacion 2-aria sobre $\mathbf{R}$

(E2) Hay exactamente dos relaciones 0-arias sobre A, a saber: $\emptyset$ y $\{\Diamond\}$.

(E3) Sea $R = \{(x_1, x_2, x_3, x_4, x_5) \in \mathbf{N}^5 : x_5 = x_4\}$. Entonces R es una relacion 5-aria sobre $\mathbf{N}$. Notese que tambien R es una relacion 5-aria sobre $\mathbf{R}$

(E4) $\emptyset$ es una relacion n-aria sobre A, cualesquiera sea $n \in \omega$ y A.

1.9 Funciones Σ-mixtas

Sea Σ un alfabeto finito. Dados $n, m \in \omega$, usaremos $\omega^n \times \Sigma^{*m}$ para abreviar la expresion

$$\overbrace{\omega \times ... \times \omega}^{n \text{ veces}} \times \overbrace{\Sigma^* \times ... \times \Sigma^*}^{m \text{ veces}}$$

Por ejemplo, $\omega^3 \times \Sigma^{*4}$ sera una forma abreviada de escribir $\omega \times \omega \times \omega \times \Sigma^* \times \Sigma^* \times \Sigma^* \times \Sigma^*$. Debe quedar claro que estamos haciendo cierto abuso notacional ya que en principio si no hacemos esta convencion notacional, $\omega^3 \times \Sigma^{*4}$ denota un conjunto de pares y $\omega \times \omega \times \omega \times \Sigma^* \times \Sigma^* \times \Sigma^* \times \Sigma^*$ es un conjunto de 7-uplas.

Notese que cuando $n = m = 0$, tenemos que $\omega^n \times \Sigma^{*m}$ denota el conjunto $\{\Diamond\}$ y si $m = 0$, entonces $\omega^n \times \Sigma^{*m}$ denota el conjunto ω^n.

Con esta convencion notacional, un elemento generico de $\omega^n \times \Sigma^{*m}$ es una $(n+m)$-upla de la forma $(x_1, ..., x_n, \alpha_1, ..., \alpha_m)$. Para abreviar, escribiremos $(\vec{x}, \vec{\alpha})$ en lugar de $(x_1, ..., x_n, \alpha_1, ..., \alpha_m)$.

Definicion de funcion Σ-mixta Sea Σ un alfabeto finito. Dada una funcion f, diremos que f es Σ-*mixta* si cumple las siguientes propiedades

(M1) Existen $n, m \geq 0$, tales que $D_f \subseteq \omega^n \times \Sigma^{*m}$

(M2) Ya sea $I_f \subseteq \omega$ o $I_f \subseteq \Sigma^*$

Algunos ejemplos:

E_1 Sea $\Sigma = \{\square, \%, \blacktriangle\}$. La funcion

$$\begin{array}{rcl} f : \omega \times \{\square, \%, \blacktriangle\}^* & \rightarrow & \omega \\ (x, \alpha) & \rightarrow & x + |\alpha| \end{array}$$

es Σ-mixta ya que se cumple (M1) con $n = m = 1$ y (M2). Notese que f no es $\{\square, \%\}$-mixta ya que no cumple (M1) respecto del alfabeto $\{\square, \%\}$. Sin envargo note que f es $\{\square, \%, \blacktriangle, @\}$-mixta

E_2 La funcion

$$\begin{array}{rcl} \omega^4 & \rightarrow & \omega \\ (x, y, z, w) & \rightarrow & x + y \end{array}$$

es Σ-mixta cualesquiera sea el alfabeto Σ

E_3 Sea $\Sigma = \{\square, @\}$. La funcion

$$\begin{array}{rcl} \{\square\square\square, @@\} & \rightarrow & \omega \\ \alpha & \rightarrow & |\alpha| \end{array}$$

es Σ-mixta ya que se cumple (M1) (con $n = 0$ y $m = 1$) y (M2)

E_4 Supongamos $\Sigma = \emptyset$. Tenemos entonces que $\Sigma^* = \{\varepsilon\}$. Por ejemplo

$$\begin{array}{rcl} D & \rightarrow & \omega \\ (x, \varepsilon, \varepsilon, \varepsilon) & \rightarrow & x^2 \end{array}$$

donde $D = \{(x, \varepsilon, \varepsilon, \varepsilon) : x \text{ es impar}\}$, es Σ-mixta (con $n = 1$ y $m = 3$ en (M1)). Tambien notese que

$$\begin{array}{rcl} \{(\varepsilon, \varepsilon)\} & \rightarrow & \{\varepsilon\} \\ (\varepsilon, \varepsilon) & \rightarrow & \varepsilon \end{array}$$

es Σ-mixta.

Dejamos al lector la facil prueba del siguiente resultado basico.

Lemma 8 *Supongamos $\Sigma \subseteq \Gamma$ son alfabetos finitos. Entonces si f es una funcion Σ-mixta, f es Γ-mixta*

Una funcion Σ-mixta f es Σ-*total* cuando haya $n, m \in \omega$ tales que $D_f = \omega^n \times \Sigma^{*m}$. El lema anterior nos dice que si $\Sigma \subseteq \Gamma$, entonces toda funcion Σ-mixta es Γ-mixta. Sin envargo una funcion puede ser Σ-total y no ser Γ-total, cuando $\Sigma \subseteq \Gamma$. Por ejemplo tomemos $\Sigma = \{\square, \%, \blacktriangle\}$ y $\Gamma = \{\square, \%, \blacktriangle, !\}$, y consideremos la funcion

$$\begin{array}{rcl} f : \omega \times \Sigma^* & \rightarrow & \omega \\ (x, \alpha) & \rightarrow & x + |\alpha| \end{array}$$

Es claro que f es Σ-mixta y Σ-total. Tambien es Γ-mixta ya que $D_f \subseteq \omega \times \Gamma^*$ y $I_f \subseteq \omega$, por lo cual cumple (M1) y (M2). Sin envargo f no es Γ-total ya que D_f no es igual a $\omega^n \times \Gamma^{*m}$, cualesquiera sean n y m.

Como hemos visto recien, una funcion f puede ser Σ-mixta y Γ-mixta para dos alfabetos distintos Σ y Γ e incluso es facil construir un ejemplo en el cual Σ y Γ sean incomparables como conjuntos, es decir que ninguno incluya al otro. Dejamos al lector convencerse de que si f es una funcion que es Σ-mixta para algun alfabeto Σ, entonces hay un alfabeto Σ_0 el cual es el menor de todos los alfabetos respecto de los cuales f es mixta, es decir Σ_0 cumple que f es Σ_0-mixta y si Γ es tal que f es Γ-mixta, entonces $\Sigma_0 \subseteq \Gamma$.

A continuacion daremos algunas funciones Σ-mixtas basicas las cuales seran frecuentemente usadas.

1.9.1 Funciones *Suc* y *Pred*

La *funcion sucesor* es definida por

$$\begin{array}{rcl} Suc : \omega & \rightarrow & \omega \\ n & \rightarrow & n + 1 \end{array}$$

La *funcion predecesor* es definida por

$$\begin{array}{rcl} Pred : \mathbf{N} & \rightarrow & \omega \\ n & \rightarrow & n - 1 \end{array}$$

1.9.2 Las funciones d_a

Sea Σ un alfabeto no vacio. Para cada $a \in \Sigma$, definamos

$$\begin{array}{rcl} d_a : \Sigma^* & \rightarrow & \Sigma^* \\ \alpha & \rightarrow & \alpha a \end{array}$$

La funcion d_a es llamada la funcion *derecha sub a*, respecto del alfabeto Σ.

1.9.3 Las funciones $p_i^{n,m}$

Sea Σ un alfabeto. Para $n,m,i \in \omega$ tales que $1 \leq i \leq n$, definamos

$$\begin{array}{rcl} p_i^{n,m} : \omega^n \times \Sigma^{*m} & \rightarrow & \omega \\ (\vec{x},\vec{\alpha}) & \rightarrow & x_i \end{array}$$

Para $n,m,i \in \omega$ tales que $n+1 \leq i \leq n+m$, definamos

$$\begin{array}{rcl} p_i^{n,m} : \omega^n \times \Sigma^{*m} & \rightarrow & \Sigma^* \\ (\vec{x},\vec{\alpha}) & \rightarrow & \alpha_{i-n} \end{array}$$

Las funciones $p_i^{n,m}$ son llamadas *proyecciones*. La funcion $p_i^{n,m}$ es llamada la *proyeccion* n,m,i, respecto del alfabeto Σ. Notese que esta definicion requiere que $n+m \geq 1$ ya que i debe cumplir $1 \leq i \leq n+m$.

1.9.4 Las funciones $C_k^{n,m}$ y $C_\alpha^{n,m}$

Sea Σ un alfabeto. Para $n,m,k \in \omega$, y $\alpha \in \Sigma^*$, definamos

$$\begin{array}{rcl} C_k^{n,m} : \omega^n \times \Sigma^{*m} & \rightarrow & \omega \\ (\vec{x},\vec{\alpha}) & \rightarrow & k \end{array} \qquad \begin{array}{rcl} C_\alpha^{n,m} : \omega^n \times \Sigma^{*m} & \rightarrow & \Sigma^* \\ (\vec{x},\vec{\alpha}) & \rightarrow & \alpha \end{array}$$

Notese que $C_k^{0,0} : \{\Diamond\} \rightarrow \{k\}$ y que $C_\alpha^{0,0} : \{\Diamond\} \rightarrow \{\alpha\}$.

1.9.5 La funcion pr

Definamos

$$\begin{array}{rcl} pr : \mathbf{N} & \rightarrow & \omega \\ n & \rightarrow & n\text{-esimo numero primo} \end{array}$$

Notese que $pr(1) = 2$, $pr(2) = 3$, $pr(3) = 5$, etc.

1.9.6 El tipo de una funcion mixta

Dada una funcion Σ-mixta f, si $n,m \in \omega$ son tales que $D_f \subseteq \omega^n \times \Sigma^{*m}$ y ademas $I_f \subseteq \omega$, entonces diremos que f *es una funcion de tipo* $(n,m,\#)$. Similarmente si $n,m \in \omega$ son tales que $D_f \subseteq \omega^n \times \Sigma^{*m}$ y ademas $I_f \subseteq \Sigma^*$, entonces diremos que f *es una funcion de tipo* $(n,m,*)$. Notese que si $f \neq \emptyset$, entonces hay unicos $n,m \in \omega$ y $s \in \{\#,*\}$ tales que f es una funcion de tipo (n,m,s). Sin envargo $\emptyset$ es una funcion de tipo (n,m,s) cualesquiera sean $n,m \in \omega$ y $s \in \{\#,*\}$. De esta forma, cuando $f \neq \emptyset$ hablaremos de "el tipo de f" para refererirnos a esta unica terna (n,m,s). Notese que Suc es de tipo $(1,0,\#)$ y d_a es de tipo $(0,1,*)$.

Tambien notese que la relacion "f es una funcion de tipo (n,m,s)" no depende del alfabeto Σ (que significa esto?).

1.9.7 Funciones con imagen contenida en $\omega^n \times \Sigma^{*m}$

Supongamos que $k,l,n,m \in \omega$ y que $F : D_F \subseteq \omega^k \times \Sigma^{*l} \rightarrow \omega^n \times \Sigma^{*m}$. Supongamos ademas que $n+m \geq 1$. Entonces denotaremos con $F_{(i)}$ a la funcion $p_i^{n,m} \circ F$. Notar que

$$\begin{array}{lll} F_{(i)} & : & D_F \subseteq \omega^k \times \Sigma^{*l} \rightarrow \omega \text{, para cada } i = 1,...,n \\ F_{(i)} & : & D_F \subseteq \omega^k \times \Sigma^{*l} \rightarrow \Sigma^* \text{, para cada } i = n+1,...,n+m \end{array}$$

Por lo cual cada una de las funciones $F_{(i)}$ son Σ-mixtas. Ademas notese que

$$F = [F_{(1)}, ..., F_{(n+m)}]$$

1.9.8 Predicados Σ-mixtos

Un *predicado Σ-mixto* es una funcion f la cual es Σ-mixta y ademas cumple que $I_f \subseteq \{0,1\}$. Por ejemplo

$$\begin{array}{rcl} \omega \times \omega & \rightarrow & \omega \\ (x,y) & \rightarrow & \left\{ \begin{array}{l} 1 \text{ si } x = y \\ 0 \text{ si } x \neq y \end{array} \right. \end{array} \qquad \begin{array}{rcl} \{1,2,3,4,5\} \times \Sigma^* & \rightarrow & \omega \\ (x,\alpha) & \rightarrow & \left\{ \begin{array}{l} 1 \text{ si } x = |\alpha| \\ 0 \text{ si } x \neq |\alpha| \end{array} \right. \end{array}$$

Operaciones logicas entre predicados Dados predicados $P : S \subseteq \omega^n \times \Sigma^{*m} \rightarrow \{0,1\}$ y $Q : S \subseteq \omega^n \times \Sigma^{*m} \rightarrow \{0,1\}$, con el mismo dominio, definamos nuevos predicados $(P \vee Q)$, $(P \wedge Q)$ y $\neg P$ de la siguiente manera

$$\begin{array}{rcl} (P \vee Q) : S & \rightarrow & \omega \\ (\vec{x},\vec{\alpha}) & \rightarrow & \left\{ \begin{array}{ll} 1 & \text{si } P(\vec{x},\vec{\alpha}) = 1 \text{ o } Q(\vec{x},\vec{\alpha}) = 1 \\ 0 & \text{caso contrario} \end{array} \right. \end{array}$$

$$\begin{array}{rcl} (P \wedge Q) : S & \rightarrow & \omega \\ (\vec{x},\vec{\alpha}) & \rightarrow & \left\{ \begin{array}{ll} 1 & \text{si } P(\vec{x},\vec{\alpha}) = 1 \text{ y } Q(\vec{x},\vec{\alpha}) = 1 \\ 0 & \text{caso contrario} \end{array} \right. \end{array}$$

$$\begin{array}{rcl} \neg P : S & \rightarrow & \omega \\ (\vec{x},\vec{\alpha}) & \rightarrow & \left\{ \begin{array}{ll} 1 & \text{si } P(\vec{x},\vec{\alpha}) = 0 \\ 0 & \text{si } P(\vec{x},\vec{\alpha}) = 1 \end{array} \right. \end{array}$$

1.9.9 Familias Σ-indexadas de funciones

Dado un alfabeto Σ, una *familia Σ-indexada de funciones* sera una funcion $\mathcal{G}$ tal que $D_{\mathcal{G}} = \Sigma$ y para cada $a \in D_{\mathcal{G}}$ se tiene que $\mathcal{G}(a)$ es una funcion. Algunos ejemplos:

E$_1$ Sea $\mathcal{G}$ dada por

$$\begin{array}{rcl} \mathcal{G} : \{\square, \%, \blacktriangle\} & \rightarrow & \{Suc, Pred\} \\ \square & \rightarrow & Suc \\ \% & \rightarrow & Suc \\ \blacktriangle & \rightarrow & Pred \end{array}$$

Claramente $\mathcal{G}$ es una familia $\{\square, \%, \blacktriangle\}$-indexada de funciones. Notar que

$$\mathcal{G} = \{(\square, Suc), (\%, Suc), (\blacktriangle, Pred)\}$$

Se tiene tambien por ejemplo que $\mathcal{G}(\%) = Suc$ por lo cual tambien es cierto que $\mathcal{G}(\%)(22) = 23$, etc.

E_2 Si Σ es un alfabeto no vacio, la funcion

$$\begin{array}{rcl} \mathcal{G}:\Sigma & \rightarrow & \{f : f \text{ es una funcion de } \Sigma^* \text{ en } \Sigma^*\} \\ a & \rightarrow & d_a \end{array}$$

es una familia Σ-indexada de funciones. Notar que

$$\mathcal{G} = \{(a, d_a) : a \in \Sigma\}$$

E_3 $\emptyset$ es una flia $\emptyset$-indexada de funciones

Si $\mathcal{G}$ es una familia Σ-indexada de funciones, entonces para $a \in \Sigma$, escribiremos $\mathcal{G}_a$ en lugar de $\mathcal{G}(a)$.

1.10 Conjuntos Σ-mixtos

Un conjunto S es llamado Σ-*mixto* si hay $n, m \in \omega$ tales que $S \subseteq \omega^n \times \Sigma^{*m}$. Por ejemplo,

$$\{(x, \alpha) \in \omega \times \{\blacktriangle, !\}^* : |\alpha| = x\}$$

$$\{(0, \blacktriangle\blacktriangle\blacktriangle, \varepsilon), (1, \%\blacktriangle\%, \blacktriangle\blacktriangle)\}$$

son conjuntos $\{\blacktriangle, \%, !\}$-mixtos. Tambien notese que $\emptyset$ y $\{\Diamond\}$ son conjuntos Σ-mixtos, cualesquiera sea el alfabeto Σ. Por ultimo el conjunto

$$\{(x, \varepsilon, \varepsilon, \varepsilon) : x \in \omega \text{ y } x \text{ es impar}\}$$

es $\emptyset$-mixto (con $n = 1$ y $m = 3$).

1.10.1 El tipo de un conjunto mixto

Dado un conjunto Σ-mixto S, si $n, m \in \omega$ son tales que $S \subseteq \omega^n \times \Sigma^{*m}$, entonces diremos que *$S$ es un conjunto de tipo* (n, m). Notese que si $S \neq \emptyset$, entonces hay unicos $n, m \in \omega$ tales que S es un conjunto de tipo (n, m). De esta forma, cuando $S \neq \emptyset$ hablaremos de "el tipo de S" para refererirnos a este unico par (n, m). Tambien es importante notar que de la definicion anterior sale inmediatemante que $\emptyset$ es un conjunto de tipo (n, m) cualesquiera sean $n, m \in \omega$, por lo cual cuando hablemos de EL tipo de un comjunto deberemos estar seguros de que dicho conjunto es no vacio.

Notese que ω es de tipo $(1, 0)$ y Σ^* es de tipo $(0, 1)$.

1.10.2 Conjuntos rectangulares

Un conjunto Σ-mixto S es llamado *rectangular* si es de la forma

$$S_1 \times ... \times S_n \times L_1 \times ... \times L_m,$$

con cada $S_i \subseteq \omega$ y cada $L_i \subseteq \Sigma^*$. Notar que todo subconjunto de ω es rectangular (es el caso $n = 1$ y $m = 0$). Tambien $\{\Diamond\}$ es rectangular (es el caso $n = m = 0$). Otros ejemplos:

- $\mathbf{N} \times \{1, 2\} \times \{@@, \varepsilon\}$ es rectangular (aqui $n = 2$ y $m = 1$)
- $\{!!!, !!\} \times \{@@, \varepsilon\}$ es rectangular (aqui $n = 0$ y $m = 2$)

Tambien notese que $\emptyset = \emptyset \times \emptyset$ por lo cual $\emptyset$ es un conjunto rectangular.

El concepto de conjunto rectangular es muy importante en nuestro enfoque. Aunque en general no habra restricciones acerca del dominio de las funciones y predicados, nuestra filosofia sera tratar en lo posible que los dominios de las funciones que utilicemos para hacer nuestro analisis de recursividad de los distintos paradigmas, sean rectangulares. Aunque en principio puede pareser que todos los conjuntos son rectangulares, el siguiente lema mostrara cuan ingenua es esta vision.

Lemma 9 *Sea $S \subseteq \omega \times \Sigma^*$. Entonces S es rectangular si y solo si se cumple la siguiente propiedad:*

(R) Si $(x, \alpha), (y, \beta) \in S$, entonces $(x, \beta) \in S$

Proof. Ejercicio. ■

Supongamos $\Sigma = \{\#, \blacktriangle, \%\}$. Notese que podemos usar el lema anterior para probar por ejemplo que los siguientes conjuntos no son rectangulares

- $\{(0, \#\#), (1, \%\%\%)\}$
- $\{(x, \alpha) : |\alpha| = x\}$

Dejamos como ejercicio para el lector enunciar un lema analogo al anterior pero que caracterice cuando $S \subseteq \omega^2 \times \Sigma^{*3}$ es rectangular.

1.11 Notacion lambda

Usaremos la notacion lambda de Church en la forma que se explica a continuacion. Esta notacion siempre depende de un alfabeto finito previamente fijado. En general en nuestro lenguaje matematico utilizamos diversas expresiones las cuales involucran variables que una vez fijadas en sus valores hacen que la expresion tambien represente un determinado valor

En el contexto de la notacion lambda solo se podran utilizar expresiones con caracteristicas muy especiales por lo cual a continuacion iremos describiendo que condiciones tienen que cumplir las expresiones para que puedan ser usadas en la notacion lambda

(1) Solo utilizaremos expresiones que involucran variables numericas, las cuales se valuaran en numeros de ω, y variables alfabeticas, las cuales se valuaran en palabras del alfabeto previamente fijado. Las variables numericas seran seleccionadas de la lista

$$\begin{array}{l} x, y, z, w, n, m, k, ... \\ x_1, x_2, ... \\ y_1, y_2, ... \\ etc \end{array}$$

Las variables alfabeticas seran seleccionadas de la lista

$$\begin{array}{l} \alpha, \beta, \gamma, \eta, ... \\ \alpha_1, \alpha_2, ... \\ \beta_1, \beta_2, ... \\ etc \end{array}$$

(2) Por ejemplo la expresion:

$$x + y + 1$$

tiene dos variables numericas x e y (y ninguna alfabetica). Si le asignamos a x el valor 2 y a y el valor 45, entonces la expresion $x + y + 1$ produce o representa el valor $48 = 2 + 45 + 1$.

(3) Otro ejemplo, consideremos la expresion

$$|\alpha\beta| + |\alpha|^x$$

la cual tiene una variable numerica x y dos variables alfabeticas α y β. Supongamos ademas que el alfabeto previamente fijado es $\{@, \%\}$. Si le asignamos a x el valor 2, a α el valor @@ y a β el valor %%%, entonces la expresion $|\alpha\beta| + |\alpha|^x$ produce o representa el valor $|@@\%\%\%| + |@@|^2 = 9$.

(4) Para ciertas valuaciones de sus variables la expresion puede no estar definida. Por ejemplo la expresion

$$Pred(|\alpha|)$$

no asume valor o no esta definida cuando el valor asignado a α es ε. Otro ejemplo, consideremos la expresion

$$x/(y - |\alpha|)^2$$

Esta expresion no esta definida o no asume valor para aquellas asignaciones de valores a sus variables en las cuales el valor asignado a y sea igual a la longitud del valor asignado a α.

(5) En los ejemplos anteriores las expresiones producen valores numericos pero tambien trabajaremos con expresiones que producen valores alfabeticos. Por ejemplo la expresion

$$\beta^y$$

tiene una variable numerica, y, una variable alfabetica, β, y una vez valuadas estas variables produce un valor alfabetico, a saber el resultado de elevar el valor asignado a la variable β, a el valor asignado a y.

(6) Una expresion E para poder ser utilizada en la notacion lambda relativa a un alfabeto Σ debera cumplir alguna de las dos siguientes propiedades

  (a) los valores que asuma E cuando hayan sido asignados valores de ω a sus variables numericas y valores de Σ^* a sus variables alfabeticas deberan ser siempre elementos de ω

  (b) los valores que asuma E cuando hayan sido asignados valores de ω a sus variables numericas y valores de Σ^* a sus variables alfabeticas deberan ser siempre elementos de Σ^*.

(7) Por ejemplo la expresion

$$x/2$$

no cumple la propiedad dada en (6) ya que para ciertos valores de ω asignados a la variable x, la expresion da valores que se salen de ω por lo cual no puede cumpir ni a. ni b.

(8) Otro ejemplo, si el alfabeto fijado es $\Sigma = \{@, \%\}$, entonces la expresion

$$@^x\$^y$$

no cumple la propiedad dada en (6) ya que por ejemplo cuando le asignamos a x el valor 2 y a y el valor 6, la expresion nos da la palabra @@$$$$$$ la cual no pertenece a Σ^* por lo cual no puede cumpir ni a. ni b.

(9) No necesariamente las expresiones que usaremos en la notacion lambda deben ser hechas como combinacion de operaciones matematicas conocidas. Muchas veces usaremos expresiones que involucran incluso lenguaje coloquial castellano. Por ejemplo la expresion

el menor numero primo que es mayor que x

Es claro que esta expresion para cada valor de ω asignado a la variable x produce o representa un valor concreto de ω. Otro ejemplo:

el tercer simbolo de α

notese que esta expresion, una ves fijado un alfabeto Σ, estara definida o producira un valor solo cuando le asignamos a α una palabra de Σ^* de longitud mayor o igual a 3.

(10) **Expresiones Booleanas.** A las expresiones Booleanas tales como

$$x = y + 1 \text{ y } |\alpha| \leq 22$$

las pensaremos que asumen valores del conjunto $\{0, 1\} \subseteq \omega$. Por ejemplo la expresion anterior asume o produce el valor 1 cuando le asignamos a x el valor 11, a y el valor 10 y a α la palabra ε. Las expresiones Booleanas pensadas de esta forma podran ser utilizadas en la notacion lambda si es que tambien cumplen con las anteriores condiciones.

Definicion de $\lambda x_1...x_n\alpha_1...\alpha_m [E]$

Supongamos ya hemos fijado un alfabeto finito Σ y supongamos E es una expresion la cual tiene las caracteristicas descriptas anteriormente. Sea $x_1, ..., x_n, \alpha_1, ..., \alpha_m$ una lista de variables todas distintas tal que las variables numericas que ocurren en E estan todas contenidas en la lista $x_1, ..., x_n$ y las variables alfabeticas que ocurren en E estan en la lista $\alpha_1, ..., \alpha_m$ (puede suceder que haya variables de la lista $x_1, ..., x_n, \alpha_1, ..., \alpha_m$ las cuales no ocurran en E). Entonces

$$\lambda x_1...x_n\alpha_1...\alpha_m [E]$$

denotara la funcion definida por:

(L1) El dominio de $\lambda x_1...x_n\alpha_1...\alpha_m [E]$ es el conjunto de las $(n+m)$-uplas $(k_1, ..., k_n, \beta_1, ..., \beta_m) \in \omega^n \times \Sigma^{*m}$ tales que E esta definida cuando le asignamos a cada x_i el valor k_i y a cada α_i el valor β_i.

(L2) $\lambda x_1...x_n\alpha_1...\alpha_m [E] (k_1, ..., k_n, \beta_1, ..., \beta_m) =$ valor que asume o representa E cuando le asignamos a cada x_i el valor k_i y a cada α_i el valor β_i.

Notese que por tener E la propiedad (6) de mas arriba, la funcion $\lambda x_1...x_n\alpha_1...\alpha_m [E]$ es Σ-mixta de tipo (n, m, s) para algun $s \in \{\#, *\}$. Algunos ejemplos:

(a) Supongamos fijamos el alfabeto $\Sigma = \{@, ?, ¡\}$. Entonces $\lambda x\alpha \left[\alpha^{2x}\right]$ es la funcion

$$\begin{array}{rcl} \omega \times \{@, ?, ¡\}^* & \to & \{@, ?, ¡\}^* \\ (x, \alpha) & \to & \alpha^{2x} \end{array}$$

Aqui el lector puede notar la dependencia de la notacion lambda respecto del alfabeto fijado. Si en lugar de fijar $\Sigma = \{@, ?, ¡\}$ hubieramos fijado $\Sigma = \{\%\}$, entonces $\lambda x\alpha \left[\alpha^{2x}\right]$ denotaria otra funcion, a saber

$$\begin{array}{rcl} \omega \times \{\%\}^* & \to & \{\%\}^* \\ (x, \alpha) & \to & \alpha^{2x} \end{array}$$

(b) Supongamos fijamos el alfabeto $\Sigma = \{@, ?, ¡\}$. Entonces $\lambda xyz\alpha \left[\alpha^{2x}\right]$ es la funcion

$$\begin{array}{rcl} \omega^3 \times \{@, ?, ¡\}^* & \rightarrow & \{@, ?, ¡\}^* \\ (x, y, z, \alpha) & \rightarrow & \alpha^{2x} \end{array}$$

(c) Supongamos fijamos el alfabeto $\Sigma = \{\%, !\}$. Entonces $\lambda\alpha\beta\left[\alpha\beta\right]$ es la funcion

$$\begin{array}{rcl} \{\%, !\}^* \times \{\%, !\}^* & \rightarrow & \{\%, !\}^* \\ (\alpha, \beta) & \rightarrow & \alpha\beta \end{array}$$

Tambien tenemos que $\lambda\beta\alpha\left[\alpha\beta\right]$ es la funcion

$$\begin{array}{rcl} \{\%, !\}^* \times \{\%, !\}^* & \rightarrow & \{\%, !\}^* \\ (\beta, \alpha) & \rightarrow & \alpha\beta \end{array}$$

Notese que estas funciones son distintas. Por ejemplo $\lambda\alpha\beta\left[\alpha\beta\right](\%, !) = \%!$ y $\lambda\beta\alpha\left[\alpha\beta\right](\%, !) = !\%$

(d) Independientemente de quien sea Σ el alfabeto previamente fijado, tenemos que $\lambda xy[x + y]$ es la funcion

$$\begin{array}{rcl} \omega^2 & \rightarrow & \omega \\ (x, y) & \rightarrow & x + y \end{array}$$

Tambien $\lambda xyzw[x + w]$ es la funcion

$$\begin{array}{rcl} \omega^4 & \rightarrow & \omega \\ (x, y, z, w) & \rightarrow & x + w \end{array}$$

(e) Supongamos fijamos el alfabeto $\Sigma = \{@, ?, ¡\}$. Entonces por la clausula (L1) tenemos que el dominio de la funcion $\lambda xy\alpha\beta\left[Pred(|\alpha|) + Pred(y)\right]$ es

$$D = \left\{(x, y, \alpha, \beta) \in \omega^2 \times \Sigma^{*2} : |\alpha| \geq 1 \text{ y } y \geq 1\right\}$$

Es decir que $\lambda xy\alpha\beta\left[Pred(|\alpha|) + Pred(y)\right]$ es la funcion

$$\begin{array}{rcl} D & \rightarrow & \omega \\ (x, y, \alpha, \beta) & \rightarrow & Pred(|\alpha|) + Pred(y) \end{array}$$

(f) Atentos a (10) de mas arriba, la funcion $\lambda xy\left[x = y\right]$ es el predicado

$$\begin{array}{rcl} \omega \times \omega & \rightarrow & \omega \\ (x, y) & \rightarrow & \left\{ \begin{array}{l} 1 \text{ si } x = y \\ 0 \text{ si } x \neq y \end{array} \right. \end{array}$$

y $\lambda x\alpha\left[Pred(x) = |\alpha|\right]$ es el predicado

$$\begin{array}{rcl} \mathbf{N} \times \Sigma^* & \rightarrow & \omega \\ (x, \alpha) & \rightarrow & \left\{ \begin{array}{l} 1 \text{ si } Pred(x) = |\alpha| \\ 0 \text{ si } Pred(x) \neq |\alpha| \end{array} \right. \end{array}$$

Tambien $\lambda\alpha\beta\,[\alpha = \beta]$ es el predicado

$$\begin{array}{rcl} \Sigma^* \times \Sigma^* & \rightarrow & \omega \\ (\alpha,\beta) & \rightarrow & \left\{ \begin{array}{l} 1 \text{ si } \alpha = \beta \\ 0 \text{ si } \alpha \neq \beta \end{array} \right. \end{array}$$

(g) Notar que para $S \subseteq \omega^n \times \Sigma^{*m}$ se tiene que $\chi_S^{\omega^n \times \Sigma^{*m}} = \lambda x_1...x_n\alpha_1...\alpha_m\,[(\vec{x},\vec{\alpha}) \in S]$

(h) Como dijimos, la notacion lambda depende del alfabeto previamnete fijado, aunque para el caso en que la lista de variables que sigue a la letra λ no tenga variables alfabeticas, la funcion representada no depende del alfabeto

@@finpagina@@

1.12 Ordenes naturales sobre Σ^*

En esta seccion daremos biyecciones naturales entre Σ^* y ω, para cada alfabeto no vacio Σ. Dichas biyecciones dependen de tener asociado a Σ un orden total. Primero haremos un caso particular pero que tiene un interes extra ya que esta emparentado con nuestra notacion decimal clasica de los numeros de ω.

1.12.1 Notacion decimal sin 0

Llamaremos *numerales* a los siguientes simbolos

$$0\ 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9$$

Usaremos Num para denotar el conjunto de numerales. Notese que $Num \cap \omega = \emptyset$. Es decir, no debemos confundir los simbolos que usualmente denotan los primeros diez numeros enteros con los numeros que ellos denotan. De hecho en china o japon los primeros diez numeros enteros se denotan con otros simbolos. Similarmente las palabras pertenecientes a Num^* denotan (notacion decimal) a los numeros de ω pero debemos tener en cuenta que $Num^* \cap \omega = \emptyset$. Cuando tratamos con palabras de Num^*, debemos ser cuidadosos ya que muchas veces en nuestro discurso matematico (es decir las guias, el apunte, lo que escriben los profesores en el pizarron, etc) representamos dos objetos diferentes de la misma forma. Por ejemplo 45 puede estar denotando al numero entero cuarenta y cinco o tambien 45 puede estar denotando la palabra de longitud 2 cuyo primer simbolo es el numeral 4 y cuyo segundo simbolo es el numeral 5, es decir ella misma. Por dar otro ejemplo, el simbolo 1 en nuestro discurso algunas veces se denotara a si mismo y otras veces denotara al numero uno.

Es bien conocido que, en notacion decimal, las siguientes palabras del alfabeto Num, denotan, de menor a mayor, a los numeros de ω

$$0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, ...$$

Por supuesto esta lista de palabras es infinita pero asumimos que el lector sabe como obtener la palabra siguiente a cada miembro de la lista (i.e. sumar 1 en notacion decimal), lo cual determina por completo la lista conociendo que la misma comienza con la palabra 0.

Cabe destacar que debido a la presencia del numeral 0 en la lista, la n-esima palabra representa o denota al numero $n-1$ o, dicho de otra forma, el numero $n \in \omega$ es representado por la $(n+1)$-esima palabra de la lista.

Un detalle de la representacion decimal de numeros de ω mediante palabras de Num^* es que la misma no nos da una biyeccion entre Num^* y ω ya que por ejemplo las palabras 00016 y 16 representan el mismo numero. Dicho de otra forma en la lista anterior no figuran todas las palabras de Num^*, a saber estan omitidas todas las palabras que comienzan con el simbolo 0 y tienen longitud mayor que uno. A continuacion daremos una representacion de los numeros de ω mediante palabras, la cual no tendra este problema. El alfabeto que usaremos tendra todos los numerales menos el 0 y ademas tendra un simbolo para denotar al numero diez, a saber el simbolo d. Es decir

$$\widetilde{Num} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, d\}$$

Representaremos a los numeros de ω con la siguiente lista infinita de palabras de $\widetilde{Num}$

$\varepsilon, 1, 2, 3, 4, 5, 6, 7, 8, 9, d,$
$11, 12, ..., 1d, 21, 22, ..., 2d, ..., 91, 92, ..., 9d, d1, d2, ..., dd,$
$111, 112, ..., 11d, 121, 122, ..., 12d, ...$

El lector ya se habra dado cuenta de que el siguiente a una palabra α de la lista anterior se obtiene aplicando las siguientes clausulas

- C_1 si $\alpha = d^n$, con $n \geq 0$ entonces el siguiente de α es 1^{n+1}
- C_2 si α no es de la forma d^n, con $n \geq 0$, entonces el siguiente de α se obtiene de la siguiente manera:
  - (a) buscar de derecha a izquierda el primer simbolo no igual a d
  - (b) reemplazar dicho simbolo por su siguiente en la lista $1, 2, 3, 4, 5, 6, 7, 8, 9, d$
  - (c) reemplazar por el simbolo 1 a todos los simbolos iguales a d que ocurrian a la derecha del simbolo reemplazado

Notese que

- El numero 0 es representado en la lista anterior con la palabra ε
- El numero 1 es representado en la lista anterior con la palabra 1

$$\vdots$$

- El numero 9 es representado en la lista anterior con la palabra 9

- El numero 10 es representado en la lista anterior con la palabra d

- El numero 11 es representado en la lista anterior con la palabra 11

$$\vdots$$

- El numero 19 es representado en la lista anterior con la palabra 19

- El numero 20 es representado en la lista anterior con la palabra $1d$

- El numero 21 es representado en la lista anterior con la palabra 21

- El numero 22 es representado en la lista anterior con la palabra 22

$$\vdots$$

Como puede notarse en estos primeros veinte y pico numeros solo dos (el 0 y el 20) se representan en forma distinta a la reprentacion decimal clasica. Es natural que ε denote al numero 0 y ademas notese que la palabra $1d$ (que en la lista representa el 20) puede leerse como "diecidiez" (es decir la palabra que sigue a "diecinueve") que justamente es 20. Por supuesto con esta manera de pensar la palabra $2d$ deberiamos leerla como "ventidiez" y si nos fijamos en la lista ella representa al numero treinta lo cual nuevamente es muy natural. Otro ejemplo: a $6d$ deberiamos leerla como "sesentidiez" y es natural ya que en la lista representa al setenta. Tambien, la palabra $9d$ puede leerse noventidiez ya que representa en la lista al numero 100.

La lista anterior va representando los numeros de ω en forma muy natural pero aunque nuestra intuicion nos diga que no, en principio podria pasar que una misma palabra del alfabeto $\widetilde{Num}$ ocurra dos veces en la lista y esto nos diria que una misma palabra estaria representando a dos numeros distintos. Tambien, en principio podria suceder que haya una palabra del alfabeto $\widetilde{Num}$ la cual nunca figure en la lista. Mas abajo probaremos que estas dos posibilidades no suceden, es decir muestran que

(S) Toda palabra de $\widetilde{Num}^*$ aparece en la lista

(I) Ninguna palabra de $\widetilde{Num}^*$ aparece mas de una ves

Notese que la propiedad (S) nos dice que la funcion

$$\begin{array}{rcl} * : \omega & \rightarrow & \widetilde{Num}^* \\ n & \rightarrow & (n+1)\text{-esimo elemento de la lista} \end{array}$$

es sobreyectiva y la propiedad (I) nos garantiza que dicha funcion es inyectiva, por lo cual entre las dos nos garantizan que dicha representacion establece una biyeccion entre ω y $\widetilde{Num}^*$.

Por supuesto, la pregunta que inmediatamente surge es como calcular la inversa de $*$. Llamemos $\#$ a la inversa de $*$. Notese que dada una palabra $\alpha \in \widetilde{Num}^*$, el numero $\#(\alpha)$ es justamente el numero representado por la palabra α, o dicho de otra forma $\#(\alpha)$ es la posicion que ocupa α en la lista, contando desde el 0 (es decir α es la $(\#(\alpha)+1)$-esima palabra de la lista). Por ejemplo:

$$\begin{gathered}
\#(\varepsilon) = 0 \\
\#(1) = 1 \\
\vdots \\
\#(9) = 9 \\
\#(d) = 10 \\
\#(11) = 11 \\
\#(12) = 12 \\
\vdots \\
\#(19) = 19 \\
\#(1d) = 20
\end{gathered}$$

Aqui hay que tener cuidado como leemos las igualdades anteriores. Por ejemplo en la igualdad

$$\#(1) = 1$$

la primera ocurrencia del simbolo 1 se refiere al numeral uno, es decir denota una palabra y la segunda ocurrencia se esta refiriendo al numero uno, es decir denota un numero.

Dejamos al lector el ejercicio de ganar intuicion con ejemplos hasta que se convensa de que tal como en el caso de la notacion decimal, el numero $\#(\alpha)$ se expresa como una suma de potencias de 10, con los coeficientes dados en funcion de los simbolos de α. Mas concretamente si $\alpha = s_1 s_2 ... s_k$ con $k \geq 1$ y $s_1, s_2, ..., s_k \in \widetilde{Num}$, entonces

$$\#(\alpha) = \#(s_1).10^{k-1} + \#(s_2).10^{k-2} + ... + \#(s_k).10^0$$

No daremos aqui una prueba de este hecho ya que lo probaremos abajo para el caso general. Para ganar intuicion sobre el mismo el lector puede ver mas abajo la prueba de las propiedades (S) e (I), desde donde se ve con mas claridad como va aumentando la funcion $\#$ a medida que recorremos la lista de izquierda a derecha. Algunos ejemplos

$$\begin{aligned}
\#(1d) &= 1.10^1 + 10.10^0 = 10 + 10 = 20 \\
\#(dd) &= 10.10^1 + 10.10^0 = 100 + 10 = 110 \\
\#(111) &= 1.10^2 + 1.10^1 + 1.10^0 = 100 + 10 + 1 = 111 \\
\#(1d3d) &= 1.10^3 + 10.10^2 + 3.10^1 + 10.10^0
\end{aligned}$$

Ahora que sabemos que las palabras de $\widetilde{Num}$ representan los numeros como suma de potencias de diez, en forma analoga a la notacion decimal clasica, podemos refozar aun mas la analogia poniendo nombres adecuados que, tal como en el caso clasico, nos permitan leer las palabras de $\widetilde{Num}$ describiendo su suma de potencias asociada. Por ejemplo podriamos llamar "decenta" al numero 100, por analogia a "treinta", "cuarenta",...,"noventa". O sea una decenta es diez veces diez. De esta forma la palabra $d1$ se leera "decenta y uno" y esto es natural ya que en la lista representa al 101. La palabra dd se leera "decenta y diez" y esto describe a la perfeccion el numero que representa, i.e. el $10.10 + 10 = 110$. La palabra que sigue en la lista a dd es 111 la cual representa al 111, es decir aqui como en los otros casos vistos en los cuales no hay ocurrencias del simbolo d la palabra representa al mismo numero que representa en la notacion decimal clasica. Por dar otro ejemplo, la palabra $59d3$ se leera "cinco mil novecientos decenta y tres" y representara al numero 6003.

Para seguir debemos ponerle nombre a "diez veces cien", es decir, "decientos" (por analogia con "novecientos = nueve veces cien") denotara al numero $1000 = 10.100$. De esta forma la palabra $d51$ se leera "decientos cincuenta y uno" y esto es natural ya que pensando un rato se puede ver que ella representa al 1051. Tambien, la palabra ddd se leera "decientos decenta y diez" y representara al numero 1110.

Prueba de las propiedades (S) e (I) Dado que el siguiente a un elemento α de la lista es de la misma longitud que α o tiene longitud igual a $|\alpha| + 1$, podemos representar la lista anterior de la siguiente manera:

$$B_0; B_1; B_2; B_3; B_4; ...$$

donde cada B_n es, por definicion, la parte de la lista en la cual las palabras tienen longitud exactamente n. Por ejemplo:

- B_0 es ε
- B_1 es $1, 2, 3, 4, 5, 6, 7, 8, 9, d$
- B_2 es $11, 12, ..., 1d, 21, 22, ..., 2d, ..., 91, 92, ..., 9d, d1, d2, ..., dd$

Notese que hasta el momento nada nos asegura que no suceda que para algun n se de que B_n sea una lista infinita, lo cual ademas nos diria que los bloques $B_{n+1}, B_{n+2}, ...$ son todos vacios. Es decir podria pasar que la lista se estanque en una longitud n y nunca aparezca una palabra de longitud mayor que n. Esto por supuesto obligaria a que se repitan muchas veces palabras de dicha longitud n ya que hay una cantidad finita de las mismas (10^n).

Por supuesto nuestra intuicion nos dice que en el bloque B_n estan listadas sin repeticion todas las palabras de $\widetilde{Num}^*$ de longitud n, pero debemos justificar esto con argumentos solidos. Algunas propiedades basicas que se pueden probar facilmente son:

(1) Si $B_n = \alpha_1, ..., \alpha_k$, entonces $\alpha_1 = 1^n$ y $\alpha_k = d^n$

(2) Si d^n ocurre en B_n lo hace en la ultima posicion

estas propiedades son consecuencias inmediatas de como se calcula el elemento siguiente a uno dado en la lista y son dejadas como ejercicio. Otra propiedad importante es la siguiente

(3) Si $B_n = \alpha_1, ..., \alpha_k$, entonces $B_{n+1} = 1\alpha_1, ..., 1\alpha_k, 2\alpha_1, ..., 2\alpha_k, ..., d\alpha_1, ..., d\alpha_k$

Para probar (3) es muy util el siguiente resultado obvio

Lemma 10 *Sea $\sigma \in \widetilde{Num}$ y supongamos $\alpha \in \widetilde{Num}^*$ no es de la forma d^n. Entonces el siguiente a $\sigma\alpha$ es $\sigma\beta$ donde β es el siguiente a α*

Dejamos como ejercicio al lector hacer la prueba de (3) usando el lema anterior y las propiedades (1) y (2). Ahora es facil usando (3) probar inductivamente que

(4) B_n es una lista sin repeticiones de todas las palabras de longitud n

Pero claramente de (4) se desprenden en forma obvia las propiedades (S) y (I).

1.12.2 El caso general

Sea Σ un alfabeto no vacio y supongamos $\leq$ es un orden total sobre Σ. Supongamos que $\Sigma = \{a_1, ..., a_n\}$, con $a_1 < a_2 < ... < a_n$. Inspirados en la lista dada anteriormente de las palabras de $\widetilde{Num}^*$, podemos dar la siguiente lista de palabras de Σ^*

$$\begin{array}{l}
\varepsilon, a_1, a_2, ..., a_n, \\
a_1a_1, a_1a_2, ..., a_1a_n, a_2a_1, a_2a_2, ..., a_2a_n, ..., a_na_1, a_na_2, ..., a_na_n, \\
a_1a_1a_1, a_1a_1a_2, ..., a_1a_1a_n, a_1a_2a_1, a_1a_2a_2, ..., a_1a_2a_n, ..., a_1a_na_1, a_1a_na_2, a_1a_na_n, \\
a_2a_1a_1, a_2a_1a_2, ..., a_2a_1a_n, a_2a_2a_1, a_2a_2a_2, ..., a_2a_2a_n, ..., a_2a_na_1, a_2a_na_2, a_2a_na_n, \\
\qquad\qquad\qquad\qquad\vdots \\
a_na_1a_1, a_na_1a_2, ..., a_na_1a_n, a_na_2a_1, a_na_2a_2, ..., a_na_2a_n, ..., a_na_na_1, a_na_na_2, a_na_na_n, \\
a_1a_1a_1a_1, a_1a_1a_1a_2, ...
\end{array}$$

El objetivo es probar que la lista anterior enumera sin repeticiones todas las palabras de Σ^*, i.e. produce naturalmente una biyeccion entre ω y Σ^*. Pero antes debemos definir mas formalmente la lista. Para esto definamos $s^{\leq} : \Sigma^* \to \Sigma^*$ de la siguiente manera

- $s^{\leq}((a_n)^m) = (a_1)^{m+1}$, para cada $m \geq 0$
- $s^{\leq}(\alpha a_i (a_n)^m) = \alpha a_{i+1} (a_1)^m$, cada vez que $\alpha \in \Sigma^*$, $1 \leq i < n$ y $m \geq 0$

Notese que la definicion de $s^{\leq}$ es correcta ya que una palabra de Σ^* ya sea es de la forma $(a_n)^m$, con $m \geq 0$, o es de la forma $\alpha a_i(a_n)^m$, con $\alpha \in \Sigma^*$, $1 \leq i < n$ y $m \geq 0$; y estos dos casos posibles son mutuamente excluyentes.

Claramente se tiene entonces que la lista anterior puede ser escrita de la siguiente manera

$$\varepsilon, s^{\leq}(\varepsilon), s^{\leq}(s^{\leq}(\varepsilon)), s^{\leq}(s^{\leq}(s^{\leq}(\varepsilon))), s^{\leq}(s^{\leq}(s^{\leq}(s^{\leq}(\varepsilon)))), ...$$

Con esta definicion formal de la lista, podemos probar de la misma forma en la que lo hicimos arriba para el caso $\Sigma = Num$ que:

(S) Toda palabra de Σ^* aparece en la lista

(I) Ninguna palabra de Σ^* aparece mas de una ves en la lista

(dejamos al lector los detalles por tratarse de un argumento completamente similar).

Definamos $*^{\leq} : \omega \rightarrow \Sigma^*$ recursivamente de la siguiente manera:

- $*^{\leq}(0) = \varepsilon$
- $*^{\leq}(i+1) = s^{\leq}(*^{\leq}(i))$

Es claro que entonces $*^{\leq}(i)$ nos da el $(i+1)$-esimo elemento de la lista, o lo que es lo mismo, el i-esimo elemento de la lista contando desde el 0. O sea que las propiedades (S) y (I) nos garantizan que la funcion $*^{\leq}$ es biyectiva. A continuacion describiremos su inversa. Primero un lema facil pero muy importante.

Lemma 11 *Sea Σ un alfabeto no vacio y supongamos $\leq$ es un orden total sobre Σ. Supongamos que $\Sigma = \{a_1, ..., a_n\}$, con $a_1 < a_2 < ... < a_n$. Entonces para cada $\alpha \in \Sigma^* - \{\varepsilon\}$ hay unicos $k \in \omega$ y $i_0, i_1, ..., i_k \in \{1, ..., n\}$ tales que*

$$\alpha = a_{i_k} ... a_{i_0}$$

Notar que k del lema anterior es $|\alpha| - 1$ y los numeros $i_k, ..., i_0$ van dando el numero de orden de cada simbolo de α yendo de izquierda a derecha. Por ejemplo si $\Sigma = \{\%, !, @\}$ y $\leq$ es el orden total sobre Σ dado por $\% <! < @$ (es decir que aqui $a_1 = \%$, $a_2 =!$ y $a_3 = @$) entonces para la palabra !%@%@ tenemos $k = 4$ y $i_4 = 2$, $i_3 = 1$, $i_2 = 3$, $i_1 = 1$ y $i_0 = 3$. Sin envargo si hubieramos tomado el orden dado por $@ < \% <!$, para la misma palabra hubieramos tenido $i_4 = 3$, $i_3 = 2$, $i_2 = 1$, $i_1 = 2$ y $i_0 = 1$.

Ahora podemos definir la funcion $\#^{\leq}$ de la siguiente manera

$$\begin{array}{rcl} \#^{\leq} : \Sigma^* & \rightarrow & \omega \\ \varepsilon & \rightarrow & 0 \\ a_{i_k} ... a_{i_0} & \rightarrow & i_k n^k + ... + i_0 n^0 \end{array}$$

Lemma 12 *La funcion $\#^{\leq}$ es la inversa de $*^{\leq}$*

Proof. Primero probaremos por induccion en x que

(a) Para cada $x \in \omega$, se tiene que $\#^{\leq}(*^{\leq}(x)) = x$

El caso $x = 0$ es trivial. Supongamos que $\#^{\leq}(*^{\leq}(x)) = x$, veremos entonces que $\#^{\leq}(*^{\leq}(x+1)) = x+1$. Sean $k \geq 0$ y $i_k, ..., i_0$ tales que $*^{\leq}(x) = a_{i_k}...a_{i_0}$. Ya que $\#^{\leq}(*^{\leq}(x)) = x$ tenemos que $x = i_k n^k + ... + i_0 n^0$. Hay varios casos.
Caso $i_0 < n$. Entonces $*^{\leq}(x+1) = s^{\leq}(*^{\leq}(x)) = a_{i_k}...a_{i_0+1}$ por lo cual

$$\begin{aligned} \#^{\leq}(*^{\leq}(x+1)) &= i_k n^k + i_{k-1} n^{k-1} + ... + (i_0+1) n^0 \\ &= \left(i_k n^k + i_{k-1} n^{k-1} + ... + i_0 n^0\right) + 1 \\ &= x + 1 \end{aligned}$$

Caso $i_k = i_{k-1} = ... = i_0 = n$. Entonces $*^{\leq}(x+1) = s^{\leq}(*^{\leq}(x)) = (a_1)^{k+2}$ por lo cual

$$\begin{aligned} \#^{\leq}(*^{\leq}(x+1)) &= 1n^{k+1} + 1n^k + ... + 1n^1 + 1n^0 \\ &= \left(nn^k + nn^{k-1} + ... + nn^0\right) + 1 \\ &= x + 1 \end{aligned}$$

Caso $i_0 = i_1 = ... = i_h = n$, $i_{h+1} \neq n$, para algun $0 \leq h < k$. Entonces $*^{\leq}(x+1) = s^{\leq}(*^{\leq}(x)) = a_{i_k}...a_{i_{h+2}} a_{i_{h+1}+1} (a_1)^h$ por lo cual

$$\begin{aligned} \#^{\leq}(*^{\leq}(x+1)) &= i_k n^k + ... + i_{h+2} n^{h+2} + (i_{h+1}+1) n^{h+1} + 1n^h + ... + 1n^1 + 1n^0 \\ &= \left(i_k n^k + ... + i_{h+2} n^{h+2} + i_{h+1} n^{h+1} + n^{h+1} + n^h + ... + n^1\right) + 1 \\ &= \left(i_k n^k + ... + i_{h+2} n^{h+2} + i_{h+1} n^{h+1} + nn^h + ... + nn^0\right) + 1 \\ &= x + 1 \end{aligned}$$

De esta forma hemos probado (a).

Por definicion la inversa de $*^{\leq}$ es la funcion con dominio Σ^* que a una palabra α le asocia el unico $x \in \omega$ tal que $*^{\leq}(x) = \alpha$. Es decir debemos probar que

(b) $\#^{\leq}(\alpha) =$ unico $x \in \omega$ tal que $*^{\leq}(x) = \alpha$, para cada $\alpha \in \Sigma^*$

Pero (b) es una concecuencia inmediata de (a). ■

Cabe destacar que dada una palabra α, el numero $\#^{\leq}(\alpha)$ nos dice en que posicion se hubica α en la lista, es decir α es la $(\#^{\leq}(\alpha)+1)$-esima palabra de la lista.

De los desarrollos hechos se desprende el interesante resultado. Dejamos al lector la prueba como ejercicio.

Lemma 13 *Sea $n \geq 1$ fijo. Entonces cada $x \geq 1$ se escribe en forma unica de la siguiente manera:*

$$x = i_k n^k + i_{k-1} n^{k-1} + ... + i_0 n^0,$$

con $k \geq 0$ y $1 \leq i_k, i_{k-1}, ..., i_0 \leq n$.

Como hemos visto las biyecciones dadas producen una "identificacion" entre numeros de ω y palabras del alfabeto Σ. Es decir, en algun sentido identificamos palabras y numeros ya que se corresponden biunivocamente. Supongamos que α es una palabra de $\Sigma^* - \{\varepsilon\}$ y queremos "verla como un numero". Entonces en ves de ver sus simbolos vemos los ordenes de aparicion en Σ de los mismos y miramos la suma de potencias asociada.

Supongamos ahora que x es un numero de $\omega - \{0\}$ y ademas supongamos que somos super inteligentes y que cuando vemos a x vemos la secuencia unica de numeros $i_k, i_{k-1}, ..., i_0$ que nos permite expresarlo como suma de potencias segun el lema anterior. Entonces si queremos ver a x como una palabra simplemente miramos la secuencia $i_k, i_{k-1}, ..., i_0$ como palabra, reemplazando cada i_j por el simbolo i_j-esimo de Σ.

Caracter recursivo de las funciones $s^{\leq}$, $*^{\leq}$ y $\#^{\leq}$ Es un ejercicio (dejado al lector) probar que cualquiera sea $\alpha \in \Sigma^*$, se tiene que

$$\begin{aligned} s^{\leq}(\varepsilon) &= a_1 \\ s^{\leq}(\alpha a_i) &= \alpha a_{i+1}, \; i < n \\ s^{\leq}(\alpha a_n) &= s^{\leq}(\alpha) a_1 \end{aligned}$$

Notese que esto nos permite calcular recursivamente el valor de $s^{\leq}$ ya que las ecuaciones anteriores nos muestran como obtener rapidamente $s^{\leq}(\alpha a)$ en terminos de $s^{\leq}(\alpha)$ y a, donde a es un elemento cualquiera de Σ. Por supuesto, en algun momento deberemos usar el dato inicial $s^{\leq}(\varepsilon) = a_1$. En un lenguaje de programacion funcional, las tres ecuaciones anteriores son directamente un programa para computar $s^{\leq}$ o si se quiere una definicion de dicha funcion. Dejamos al lector que intente usar las ecuaciones anteriores para dar un programa imperativo que compute $s^{\leq}$ (esto esta hecho mas adelante en la primera lista de funciones Σ-efectivamente computables).

Lo mismo sucede con la funcion $*^{\leq}$ la cual fue directamente definida en forma recursiva por las ecuaciones

$$\begin{aligned} *^{\leq}(0) &= \varepsilon \\ *^{\leq}(i+1) &= s^{\leq}(*^{\leq}(i)) \end{aligned}$$

Dejamos al lector corroborar que la funcion $\#^{\leq}$ verifica las siguientes ecuaciones, las cuales obviamente pueden ser usadas para calcular recursivamente sus valores

$$\begin{aligned} \#^{\leq}(\varepsilon) &= 0 \\ \#^{\leq}(\alpha a_i) &= \#^{\leq}(\alpha).n + i \end{aligned}$$

Extension del orden total de Σ a Σ^* Podemos extender el orden de Σ a Σ^* de la siguiente manera

- $\alpha \leq \beta$ sii $\#^{\leq}(\alpha) \leq \#^{\leq}(\beta)$

Es decir $\alpha \leq \beta$ sii $\alpha = \beta$ o α ocurre antes que β en la lista. Dejamos como ejercicio para el lector probar que $\leq$ es un orden total sobre Σ^*.

Deberia ser intuitivamente claro que el orden recien definido sobre Σ^* posee las mismas propiedades matematicas que el orden usual de ω. Esto se entendera en forma mas profunda cuando veamos el concepto de isomorfismo de posets en los capitulos de logica. Veamos un ejemplo:

Lemma 14 *Si $S \subseteq \Sigma^*$ es no vacio, entonces existe $\alpha \in S$ tal que $\alpha \leq \beta$, para cada $\beta \in S$.*

1.13 Codificacion de infinituplas de numeros

Usaremos $\omega^{\mathbf{N}}$ para denotar el conjunto de todas las infinituplas con coordenadas en ω. Es decir

$$\omega^{\mathbf{N}} = \{(s_1, s_2, ...) : s_i \in \omega, \text{ para cada } i \geq 1\}.$$

Definamos el siguiente subconjunto de $\omega^{\mathbf{N}}$

$$\omega^{[\mathbf{N}]} = \left\{(s_1, s_2, ...) \in \omega^{\mathbf{N}} : \text{ hay un } n \in \mathbf{N} \text{ tal que } s_i = 0, \text{para } i \geq n\right\}.$$

Notese que $\omega^{\mathbf{N}} \neq \omega^{[\mathbf{N}]}$, por ejemplo las infinituplas

$$(10, 20, 30, 40, 50, ...)$$
$$(1, 0, 1, 0, 1, 0, 1, 0, ...)$$

no pertenecen a $\omega^{[\mathbf{N}]}$. Notese que $(s_1, s_2, ...) \in \omega^{[\mathbf{N}]}$ si y solo si solo una cantidad finita de coordenadas de $(s_1, s_2, ...)$ son no nulas (i.e. $\{i : s_i \neq 0\}$ es finito).

Definamos

$$\begin{array}{rcl} pr : \mathbf{N} & \rightarrow & \omega \\ n & \rightarrow & n\text{-esimo numero primo} \end{array}$$

Nótese que $pr(1) = 2$, $pr(2) = 3$, $pr(3) = 5$, etc.

Es bien conocido que todo numero natural es expresable como producto de primos. Por ejemplo si tomamos $x = 57596$ tenemos que $x = 2.2.7.11.11.17$. Tambien es un hecho conocido que dicha representacion en producto de primos es unica, si escribimos a los factores primos de menor a mayor, tal como lo hicimos recien con el numero 57596. El Teorema Fundamental de la Aritmetica justamente acevera esta propiedad de factorisacion unica de todo numero natural. Trataremos de escribir este teorema de una forma un poco mas "cheta".

Ya que $57596 = 2.2.7.11.11.17$, podemos escribir

$$57596 = pr(1)^2.pr(4)^1.pr(5)^2.pr(7)^1$$

Notese que ahora cada primo que interviene en la factorizacion de 57596 figura con un exponente que nos dice cuantas veces ocurre en dicha factorizacion. Hay muchos primos que no ocurren en esta factorizacion, es decir ocurren 0 veces en la misma. Pero podemos escribir

$$57596 = pr(1)^2.pr(2)^0.pr(3)^0.pr(4)^1.pr(5)^2.pr(6)^0.pr(7)^1.pr(8)^0.pr(9)^0.pr(10)^0....$$

y la igualdad no se altera ya que agregamos factores iguales a 1 (una cantidad infinita!). De esta manera cada primo interviene en la factorizacion. Ademas si vemos la infinitupla de exponentes de dicha factorizacion, es decir

$$(2, 0, 0, 1, 2, 0, 1, 0, 0, 0, ...)$$

obtenemos un elemento de $\omega^{[\mathbf{N}]}$.

Por supuesto esto lo podemos hacer con cualquier numero natural y siempre la infinitupla de exponentes sera un elemento de $\omega^{[\mathbf{N}]}$. Ademas es facil notar que estas representaciones "chetas" tambien resultan unicas.

Para probar nuestra version del Teorema Fundamental de la Aritmetica necesitaremos el siguiente lema el cual aceptaremos sin demostracion.

Lemma 15 *Si $p, p_1, ..., p_n$ son numeros primos y p divide a $p_1.p_2.\ldots.p_n$, entonces $p = p_i$, para algun i.*

Theorem 16 *Para cada $x \in \mathbf{N}$, hay una unica infinitupla $(s_1, s_2, ...) \in \omega^{[\mathbf{N}]}$ tal que*

$$x = \prod_{i=1}^{\infty} pr(i)^{s_i}$$

(Tiene sentido escribir $\prod_{i=1}^{\infty} pr(i)^{s_i}$, ya que en esta productoria solo una cantidad finita de factores son no iguales a 1.)

Proof. Primero probaremos la existencia por induccion en x. Claramente $1 = \prod_{i=1}^{\infty} pr(i)^0$, con lo cual tomando $(s_1, s_2, ...) = (0, 0, 0, ...)$ el caso $x = 1$ esta probado. Fijemos ahora un $x > 1$ y supongamos la existencia vale para cada y menor que x. Veremos que entonces vale para x. Si x es primo, entonces $x = pr(i_0)$ para algun i_0 por lo cual tenemos que $x = \prod_{i=1}^{\infty} pr(i)^{s_i}$, tomando $s_i = 0$ si $i \neq i_0$ y $s_{i_0} = 1$. Si x no es primo, entonces $x = y_1.y_2$, con $y_1, y_2 < x$. Por hipotesis inductiva tenemos que hay $(s_1, s_2, ...), (t_1, t_2, ...) \in \omega^{[\mathbf{N}]}$ tales que $y_1 = \prod_{i=1}^{\infty} pr(i)^{s_i}$ y $y_2 = \prod_{i=1}^{\infty} pr(i)^{t_i}$. Tenemos entonces que $x = \prod_{i=1}^{\infty} pr(i)^{s_i+t_i}$ lo cual concluye la prueba de la existencia.

Veamos ahora la unicidad. Suponganos que las infinituplas $(s_1, s_2, ...), (t_1, t_2, ...) \in \omega^{[\mathbf{N}]}$ son tales que

$$\prod_{i=1}^{\infty} pr(i)^{s_i} = \prod_{i=1}^{\infty} pr(i)^{t_i}$$

y ademas $s_i \neq t_i$ para algun i. Si $s_i > t_i$ entonces dividiendo ambos miembros por $pr(i)^{t_i}$ obtenemos que $pr(i)$ divide a un producto de primos todos distintos de el, lo cual es absurdo por el lema anterior. Analogamente llegamos a un absurdo si suponemos que $t_i > s_i$, lo cual nos dice que vale la unicidad. ■

Como podra notarse la existencia en el teorema anterior es facil e intuitivamente clara de probar. En realidad la potencia del Teorema Fundamental de la Aritmética radica en el hecho de que dicha factorizacion es unica.

A continuacion un poco de notacion. Dada una infinitupla $(s_1, s_2, ...) \in \omega^{[\mathbf{N}]}$ usaremos $\langle s_1, s_2, ... \rangle$ para denotar al numero $\prod_{i=1}^{\infty} pr(i)^{s_i}$. Dado $x \in \mathbf{N}$, usaremos (x) para denotar a la unica infinitupla $(s_1, s_2, ...) \in \omega^{[\mathbf{N}]}$ tal que

$$x = \langle s_1, s_2, ... \rangle = \prod_{i=1}^{\infty} pr(i)^{s_i}$$

Ademas para $i \in \mathbf{N}$, usaremos $(x)_i$ para denotar a s_i de dicha unica infinitupla. Es decir que

(1) $(x) = ((x)_1, (x)_2, ...)$

(2) $(x)_i$ es el exponente de $pr(i)$ en la (unica posible) factorizacion de x como producto de primos

Claramente entonces

(3) $\langle (x)_1, (x)_2, ... \rangle = x$, para cada $x \in \mathbf{N}$

(4) Para cada $(s_1, s_2, ...) \in \omega^{[\mathbf{N}]}$, se tiene que

$$(\langle s_1, s_2, ... \rangle)_i = s_i, \text{ para } i \in \mathbf{N}$$

Es decir que

$$(\langle s_1, s_2, ... \rangle) = (s_1, s_2, ...)$$

(Justifique con palabras las propiedades (3) y (4)). Tenemos entonces el siguiente resultado fundamental

Theorem 17 *Las funciones*

$$\begin{array}{ccc} \mathbf{N} & \rightarrow & \omega^{[\mathbf{N}]} \\ x & \rightarrow & (x) = ((x)_1, (x)_2, ...) \end{array} \qquad \begin{array}{ccc} \omega^{[\mathbf{N}]} & \rightarrow & \mathbf{N} \\ (s_1, s_2, ...) & \rightarrow & \langle s_1, s_2, ... \rangle \end{array}$$

son biyecciones una inversa de la otra.

Proof. Llamemos f a la funcion de la izquierda y g a la de la derecha. Notese que el Lema 2 nos dice que basta con probar que $f \circ g = Id_{\omega^{[\mathbf{N}]}}$ y $g \circ f = Id_{\mathbf{N}}$. Pero (3) justamente nos dice que $g \circ f = Id_{\mathbf{N}}$ y (4) nos dice que $f \circ g = Id_{\omega^{[\mathbf{N}]}}$. ■

Tal como se hace en la escuela primaria, el siguiente lema nos permite calcular $(x)_i$.

Lemma 18 *Dados $x, i \in \mathbf{N}$, se tiene que*

$$(x)_i = \max_t \left(pr(i)^t \text{ divide a } x \right)$$

Proof. Ejercicio (aplique el Lema 15). ■

Definamos la funcion $Lt : \mathbf{N} \to \omega$ de la siguiente manera:

$$Lt(x) = \begin{cases} \max_i \ (x)_i \neq 0 & \text{si } x \neq 1 \\ 0 & \text{si } x = 1 \end{cases}$$

Se tienen las siguientes propiedades basicas

Lemma 19 *Para cada $x \in \mathbf{N}$:*

1. $Lt(x) = 0$ *sii* $x = 1$
2. $x = \prod_{i=1}^{Lt(x)} pr(i)^{(x)_i}$

2 Procedimientos efectivos

Un concepto importante en ciencias de la computacion es el de *procedimiento* o *metodo* para realizar alguna tarea determinada. Nos interesan los procedimientos que estan definidos en forma precisa e inambigua, es decir aquellos en los cuales en cada paso a seguir, la tarea a realizar esta objetivamente descripta. Ademas siempre supondremos que el interprete o ejecutante es una persona.

Una caracteristica de los procedimientos que surgen en la tarea cientifica es que hay un *conjunto de datos de entrada*, es decir, el conjunto de objetos a partir de los cuales puede comenzar a realizarse el procedimiento. Tambien en los procedimientos que surgen en la tarea cientifica tenemos un *conjunto de datos de salida*, es decir el conjunto de todos los datos que el procedimiento dara como salida en alguna de las posibles ejecuciones al variar los datos de entrada posibles.

Una propiedad importante de los procedimientos es la propiedad de detencion. Si $\mathbb{P}$ es un procedimiento y d es un dato de entrada de $\mathbb{P}$, entonces diremos que $\mathbb{P}$ *se detiene partiendo de d* si en algun momento de la ejecucion de $\mathbb{P}$ partiendo de d, el ejecutante concluye una tarea encomendada por $\mathbb{P}$ y no hay una nueva tarea requerida por $\mathbb{P}$ al ejecutante. Puede pasar que para ciertos datos de entrada el procedimiento no se detenga nunca, es decir puede suceder que a medida que se vayan realizando las instrucciones o tareas, siempre el procedimiento direccione a realizar otra tarea especifica y asi sucesivamente.

Cabe destacar que en la gerga computacional a veces se dice "el procedimiento termina" y en realidad a lo que nos referimos es a que termina de realizar la tarea especifica para la cual fue diseñado e inmediatamente se detiene ya que no encomienda mas tareas para hacer. Es decir, en algun sentido terminar es mas fuerte que detenerse ya que el concepto de detencion no presupone la terminacion de ninguna tarea especifica, simplemente se refiere a que el procedimiento llega a un punto en el que no direcciona a ninguna tarea al ejecutante. De todas maneras muchas veces se usara el verbo terminar como sinonimo de detenerse.

Los procedimientos tambien deben ser repetibles, en el sentido de que si realizamos un procedimiento dos veces con el mismo dato de entrada, entonces ambas ejecuciones deben ser identicas, es decir se ralizaran las mismas tareas y en el mismo orden.

Otro aspecto muy importante a considerar es que un procedimiento puede tener pasos a seguir los cuales sean realizables solo en un sentido puramente teorico. Por ejemplo, un procedimiento puede tener una instruccion como la que se muestra a continuacion:

- si el polinomio $ax^5 + bx^4 + 421$ tiene una raiz racional, entonces realizar la tarea descripta en A, en caso contrario realizar la tarea descripta en B

$(a, b$ son datos calculados previamente). Como puede notarse mas alla de este aspecto teorico de la instruccion, su descripcion es clara y objetiva, pero en principio no es claro que se pueda ejecutar dicha instruccion en un sentido efectivo a los fines de seguir realizando las siguientes instrucciones. Un procedimiento sera llamado *efectivo* cuando cada paso del mismo sea simple y facil de realizar en forma efectiva por cualquier persona.

En general los procedimientos efectivos que nos interesan son en los que el interprete o ejecutante trabaja solo con papel y lapiz. Ademas el conjunto de datos de entrada siempre estara determinado a priori como parte de la descripcion del procedimiento. Cabe destacar que puede ser muy dificil o imposible, en general, conocer con precision el conjunto de datos de salida de un procedimiento (esto lo justificaremos mas adelante).

Tambien supondremos que los elementos de ω que intervienen en los datos de entrada y de salida estaran representados por palabras de Num usando la notacion decimal.

Quisas el procedimiento efectivo mas famoso de la matematica es aquel que se enseña en los colegios para sumar dos numeros naturales expresados en notacion decimal. Notar que el conjunto de datos de entrada de dicho procedimiento es ω^2 y el conjunto de datos de salida es el conjunto formado por todas las sumas posibles de pares de elementos de ω, es decir ω. Por supuesto este procedimiento solo usa lapiz, papel y pasos extremadamente simples a seguir en cada momento de la computacion, es decir, en algun sentido, no es necesario "entender que es lo que se esta haciendo" para llegar al final y obtener la palabra que representa en notacion decimal a la suma de los numeros iniciales. Dejamos al lector repasar este procedimiento asi como el que calcula dado un numero x no nulo de ω, al numero $x-1$, los cuales nos haran falta mas adelante en los ejemplos.

2.1 Funciones Σ-efectivamente computables

Una funcion Σ-mixta $f: D_f \subseteq \omega^n \times \Sigma^{*m} \to \omega$ sera llamada Σ-*efectivamente computable* si hay un procedimiento efectivo $\mathbb{P}$ tal que

(1) El conjunto de datos de entrada de $\mathbb{P}$ es $\omega^n \times \Sigma^{*m}$

(2) El conjunto de datos de salida esta contenido en ω.

(3) Si $(\vec{x}, \vec{\alpha}) \in D_f$, entonces $\mathbb{P}$ se detiene partiendo de $(\vec{x}, \vec{\alpha})$, dando como dato de salida $f(\vec{x}, \vec{\alpha})$.

(4) Si $(\vec{x}, \vec{\alpha}) \in (\omega^n \times \Sigma^{*m}) - D_f$, entonces $\mathbb{P}$ no se detiene partiendo desde $(\vec{x}, \vec{\alpha})$

Analogamente una funcion Σ-mixta $f: D_f \subseteq \omega^n \times \Sigma^{*m} \to \Sigma^*$ sera llamada Σ-*efectivamente computable* si hay un procedimiento efectivo $\mathbb{P}$ tal que

(1) El conjunto de datos de entrada de $\mathbb{P}$ es $\omega^n \times \Sigma^{*m}$

(2) El conjunto de datos de salida esta contenido en Σ^*.

(3) Si $(\vec{x}, \vec{\alpha}) \in D_f$, entonces $\mathbb{P}$ se detiene partiendo de $(\vec{x}, \vec{\alpha})$, dando como dato de salida $f(\vec{x}, \vec{\alpha})$.

(4) Si $(\vec{x}, \vec{\alpha}) \in (\omega^n \times \Sigma^{*m}) - D_f$, entonces $\mathbb{P}$ no se detiene partiendo desde $(\vec{x}, \vec{\alpha})$

En ambos casos diremos que $\mathbb{P}$ *computa* a la funcion f.

Notese que esta definicion para el caso $f = \emptyset$ tiene a priori cierta ambiguedad ya que cualesquiera sean $n, m \in \omega$ y $O \in \{\omega, \Sigma^*\}$ tenemos que $\emptyset : \emptyset \subseteq \omega^n \times \Sigma^{*m} \to O$ ya que $D_\emptyset = \emptyset$ y $I_\emptyset = \emptyset$. De todas maneras, cualesquiera sean los n, m y O elejidos, siempre hay un procedimiento efectivo que computa a $f = \emptyset$, i.e. un procedimiento que nunca se detiene, cualesquiera sea el dato de entrada de $\omega^n \times \Sigma^{*m}$. Es decir que la funcion $\emptyset$ es Σ-efectivamente computable cualesquiera sea el alfabeto Σ. Cabe destacar que para el caso de una funcion $f \neq \emptyset$, nuestra definicion es inambigua ya que hay unicos $n, m \in \omega$ y $O \in \{\omega, \Sigma^*\}$ tales que $f: D_f \subseteq \omega^n \times \Sigma^{*m} \to O$.

Veamos algunos ejemplos:

(E1) La funcion $\lambda xy\,[x+y]$ es Σ-efectivamente computable, cualquiera sea el alfabeto Σ ya que el procedimiento enseñado en la escuela primaria para sumar numeros en notacion decimal es efectivo y computa esta funcion. Tambien las funciones $\lambda xy\,[x.y]$ y $\lambda xy\,[x^y]$ son Σ-efectivamente computables via los procedimientos clasicos enseñados en la escuela primaria.

(E2) La funcion $C_3^{1,2}$ es Σ-efectivamente computable ya que el siguiente procedimiento $\mathbb{P}$ con conjunto de datos de entrada $\omega \times \Sigma^{*2}$ la computa:

- Independientemente de quien sea el dato de entrada $(x_1, \alpha_1, \alpha_2)$, terminar y dar como salida el numero 3

(E3) La funcion $p_3^{2,3}$ es Σ-efectivamente computable ya que el siguiente procedimiento con conjunto de datos de entrada $\omega^2 \times \Sigma^{*3}$ la computa:

- Dado el dato de entrada $(x_1, x_2, \alpha_1, \alpha_2, \alpha_3)$, terminar y dar como salida la palabra α_1

(E4) *Pred* es Σ-efectivamente computable. Para realizar el procedimiento efectivo que compute a *Pred* necesitaremos el procedimiento de la escuela primaria que dado un numero no nulo x, expresado en notacion decimal, calcula el numero $x-1$, en notacion decimal. Llamemos $\mathbb{P}_{-1}$ a dicho procedimiento. El siguiente procedimiento (con conjunto de datos de entrada igual a ω) computa a *Pred*:

Dado como dato de entrada un elemento $x \in \omega$, realizar lo siguiente:

Etapa 1

Si $x = 0$, entonces ir a Etapa 3, en caso contrario ir a Etapa 2.

Etapa 2

Correr $\mathbb{P}_{-1}$ con dato de entrada x obteniendo y como dato de salida. Dar y como dato de salida y terminar.

Etapa 3

Si $x = 0$, entonces ir a Etapa 1.

Como puede notarse el procedimiento anterior es efectivo ya que debemos entender que en la Etapa 2, los sucesivos pasos efectuados al correr $\mathbb{P}_{-1}$ son todos simples y efectivamente realizables ya que $\mathbb{P}_{-1}$ es efectivo. Por supuesto si uno quisiera ser mas prolijo, deberia reemplazar la Etapa 2 por las distintas instrucciones de $\mathbb{P}_{-1}$, referidas a x.

(E5) El predicado $\lambda xy[x<y]$ es Σ-efectivamente computable cualquiera sea el alfabeto Σ. Describiremos con palabras un procedimiento que computa a $\lambda xy[x<y]$. Su conjunto de datos de entrada es ω^2. Dado un par $(x,y) \in \omega^2$, el procedimiento primero compara las longitudes de las palabras que en notacion decimal representan a x y y. Por supuesto esto lo hace borrando de a un simbolo y viendo si alguna se termina primero. Si resultan de distinta longitud, es facil darse cuenta como sigue. En caso de que las

palabras resulten de igual longitud, entonces se hace el procedimiento clasico de ir comparando digitos de izquierda a derecha.

(E6) Veamos que la funcion $\lambda\alpha[|\alpha|]$ es Σ-efectivamente computable. Como en los lenguajes de programacion, usaremos variables y asignaciones para diseñar el procedimiento. Ademas llamemos $\mathbb{P}_{+1}$ a el procedimiento de la escuela primaria que dado un numero no nulo x, expresado en notacion decimal, calcula el numero $x+1$, en notacion decimal. Sea $\mathbb{P}$ el siguiente procedimiento.

Dado como dato de entrada un elemento $\alpha \in \Sigma^*$, realizar lo siguiente:

Etapa 1: Hacer las siguientes asignaciones

$$\begin{array}{lcl} A & \leftarrow & \alpha \\ B & \leftarrow & 0 \end{array}$$

e ir a Etapa 2.

Etapa 2: Si A no es ε, ir a Etapa 3. En caso contrario terminar y dar como salida B.

Etapa 3: Correr $\mathbb{P}_{+1}$ con dato de entrada igual al contenido de B, obteniendo y como salida. Hacer la asignacion

$$\begin{array}{lcl} A & \leftarrow & \text{resultado de remover el 1er simbolo de } A \\ B & \leftarrow & y \end{array}$$

e ir a Etapa 2.

Dejamos como ejercicio convenserse que el uso de asignaciones puede realizarse usando solo lapiz y papel. Imagine como lo haria en este ejemplo y corrobore que dicho procedimiento es efectivo y ademas computa a $\lambda\alpha[|\alpha|]$

(E7) Si $\leq$ es el orden total sobre $\Sigma = \{\blacktriangle, \%\}$ dado por $\blacktriangle < \%$, entonces veremos que la funcion $s^{\leq}$ es Σ-efectivamente computable. Por el Lema **??** tenemos que cualquiera sea $\alpha \in \Sigma^*$, se cumple

$$\begin{array}{rcl} s^{\leq}(\varepsilon) & = & \blacktriangle \\ s^{\leq}(\alpha\blacktriangle) & = & \alpha\% \\ s^{\leq}(\alpha\%) & = & s^{\leq}(\alpha)\blacktriangle \end{array}$$

Tal como lo explica dicho lema el valor de $s^{\leq}$ queda determinado por las tres ecuaciones anteriores. Usaremos esta idea para dar un procedimiento efectivo (con conjunto de datos de entrada igual a Σ^*) que compute a $s^{\leq}$.

Etapa 1: Dado el dato de entrada $\alpha \in \Sigma^*$, hacer las siguientes asignaciones

$$\begin{array}{lcl} A & \leftarrow & \alpha \\ B & \leftarrow & \varepsilon \\ F & \leftarrow & \blacktriangle \end{array}$$

e ir a Etapa 2.

Etapa 2: Si A comiensa con $\blacktriangle$, entonces hacer las siguientes asignaciones

$$\begin{array}{lll} A & \leftarrow & \text{resultado de remover el 1er simbolo de } A \\ F & \leftarrow & B\% \\ B & \leftarrow & B\blacktriangle \end{array}$$

e ir a la Etapa 2. En caso contrario ir a la Etapa 3.

Etapa 3: Si A comiensa con %, entonces hacer las siguientes asignaciones

$$\begin{array}{lll} A & \leftarrow & \text{resultado de remover el 1er simbolo de } A \\ F & \leftarrow & F\blacktriangle \\ B & \leftarrow & B\% \end{array}$$

e ir a la Etapa 2. En caso contrario ir a la Etapa 4.

Etapa 4: Dar como salida F

(E8) Usando que $s^{\leq}$ es Σ-efectivamente computable podemos ver que $*^{\leq} : \omega \rightarrow \Sigma^*$ tambien lo es ya que $*^{\leq}$ es definida con las ecuaciones

$$\begin{array}{rcl} *^{\leq}(0) & = & \varepsilon \\ *^{\leq}(x+1) & = & s^{\leq}(*^{\leq}(x)) \end{array}$$

Dejamos como ejercico para el lector diseñar procedimientos efectivos que computen las funciones:

- $\lambda xy[x \text{ divide a } y]$
- $\lambda x[pr(x)]$
- $\lambda ix[(x)_i]$

(Utilice en el diseño de los respectivos procedimientos a los procedimientos que computan las funciones $\lambda xy\,[x+y]$, $\lambda xy\,[x.y]$ y $\lambda xy\,[x^y]$)

Nota Importante: en lo que sigue muchas veces daremos procedimientos que son efectivos en terminos de otros que ya se han dado, es decir daremos un procedimiento que en principio no es claro que sea efectivo pero el cual se volveria efectivo si reemplazaramos ciertas instrucciones por la manera efectiva de simularlas. Para hacer mas dinamico el discurso no distinguiremos entre este tipo de procedimientos (efectivisables) y los efectivos propiamente dichos.

2.1.1 Constructores que preservan computabilidad efectiva

Hay muchos procesos constructivos que nos sirven para definir o construir una funcion en terminos de otras funciones dadas. Un ejemplo de esto es la composicion de funciones, la cual dadas dos funciones f, g nos permite construir su composicion, a saber $f \circ g$. Otro ejemplo es el contructor de predicados que dados dos predicados Σ-mixtos $P : S \subseteq \omega^n \times \Sigma^{*m} \to \{0, 1\}$ y $Q : S \subseteq \omega^n \times \Sigma^{*m} \to \{0, 1\}$, con el mismo dominio, nos define el predicado

$$\begin{array}{rcl} (P \vee Q) : S & \to & \omega \\ (\vec{x}, \vec{\alpha}) & \to & \left\{ \begin{array}{ll} 1 & \text{si } P(\vec{x}, \vec{\alpha}) = 1 \text{ o } Q(\vec{x}, \vec{\alpha}) = 1 \\ 0 & \text{caso contrario} \end{array} \right. \end{array}$$

Otro constructor muy importante que utilizaremos mucho es aquel que a partir de funciones $f_i : D_{f_i} \to O$, $i = 1, ..., k$, tales que $D_{f_i} \cap D_{f_j} = \emptyset$ para $i \neq j$, nos da la nueva funcion $f_1 \cup ... \cup f_k$, la cual cumple

$$\begin{array}{rcl} D_{f_1} \cup ... \cup D_{f_k} & \to & O \\ e & \to & \left\{ \begin{array}{ll} f_1(e) & \text{si } e \in D_{f_1} \\ \vdots & \vdots \\ f_k(e) & \text{si } e \in D_{f_k} \end{array} \right. \end{array}$$

Lemma 20 *Si $P : S \subseteq \omega^n \times \Sigma^{*m} \to \omega$ y $Q : S \subseteq \omega^n \times \Sigma^{*m} \to \omega$ son predicados Σ-efectivamente computables, entonces $(P \vee Q)$, $(P \wedge Q)$ y $\neg P$ lo son tambien.*

2.1.2 Composicion

Dadas funciones Σ-mixtas $f, f_1, ..., f_r$, con $r \geq 1$, diremos que la funcion $f \circ [f_1, ..., f_r]$ es *obtenida por composicion a partir de las funciones* $f, f_1, ..., f_r$. Para probar que la composicion preserva la computabilidad efectiva necesitaremos el siguiente lema.

Lemma 21 *Supongamos que $f, f_1, ..., f_r$ son funciones Σ-mixtas, con $r \geq 1$. Supongamos ademas que $f \circ [f_1, ..., f_r] \neq \emptyset$. Entonces hay $n, m, k, l \in \omega$ y $s \in \{\#, *\}$ tales que*

- *$r = n + m$*
- *f es de tipo (n, m, s)*
- *f_i es de tipo $(k, l, \#)$, para cada $i = 1, ..., n$*

- f_i *es de tipo* $(k,l,*)$, *para cada* $i=n+1,...,n+m$

Mas aun, en tal caso la funcion $f\circ[f_1,...,f_{n+m}]$ *es de tipo* (k,l,s) *y:*

$$\begin{aligned} D_{f\circ[f_1,...,f_{n+m}]} &= \left\{(\vec{x},\vec{\alpha})\in\bigcap_{i=1}^{n+m}D_{f_i}:(f_1(\vec{x},\vec{\alpha}),...,f_{n+m}(\vec{x},\vec{\alpha}))\in D_f\right\} \\ f\circ[f_1,...,f_{n+m}](\vec{x},\vec{\alpha}) &= f(f_1(\vec{x},\vec{\alpha}),...,f_{n+m}(\vec{x},\vec{\alpha})). \end{aligned}$$

Proof. Notese que $f\neq\emptyset$ y $[f_1,...,f_r]\neq\emptyset$ (por que?). Ya que $f\neq\emptyset$ tenemos que hay unicos $n,m\in\omega$ y $s\in\{\#,*\}$ tales que f es de tipo (n,m,s). Ya que $f\circ[f_1,...,f_r]\neq\emptyset$ y $I_{[f_1,...,f_r]}\subseteq I_{f_1}\times...\times I_{f_r}$, tenemos que

- $r=n+m$

- $I_{f_i}\subseteq\omega$, para cada $i=1,...,n$

- $I_{f_i}\subseteq\Sigma^*$, para cada $i=n+1,...,n+m$

Ya que $[f_1,...,f_r]\neq\emptyset$ tenemos que $D_{[f_1,...,f_r]}=\bigcap_{i=1}^{r}D_{f_i}\neq\emptyset$, por lo cual los conjuntos $D_{f_1},...,D_{f_{n+m}}$ deberan ser todos de un mismo tipo, digamos de tipo (k,l). Es decir que f_i es de tipo $(k,l,\#)$, para cada $i=1,...,n$ y f_i es de tipo $(k,l,*)$, para cada $i=n+1,...,n+m$.

Las ultimas observaciones del lema son directas de las definiciones de $[f_1,...,f_{n+m}]$ y de composicion de funciones ■

Ahora si podemos probar facilmente que se preserva la computabilidad efectiva cuando componemos

Lemma 22 *Si* $f,f_1,...,f_r$, *con* $r\geq 1$, *son* Σ*-efectivamente computables, entonces* $f\circ[f_1,...,f_r]$ *lo es.*

Proof. Si $f\circ[f_1,...,f_r]=\emptyset$, entonces claramente es Σ-efectivamente computable. Supongamos entonces que $f\circ[f_1,...,f_r]\neq\emptyset$. Por el lema anterior hay $n,m,k,l\in\omega$ y $s\in\{\#,*\}$ tales que

- $r=n+m$

- f es de tipo (n,m,s)

- f_i es de tipo $(k,l,\#)$, para cada $i=1,...,n$

- f_i es de tipo $(k,l,*)$, para cada $i=n+1,...,n+m$

Sean $\mathbb{P},\mathbb{P}_1,...,\mathbb{P}_{n+m}$ procedimientos efectivos los cuales computen las funciones $f,f_1,...,f_{n+m}$, respectivamente. Usando estos procedimientos es facil definir un procedimiento efectivo el cual compute a $f\circ[f_1,...,f_{n+m}]$. ■

2.1.3 Lema de division por casos para funciones Σ-efectivamente computables

Recordemos que si $f_i : D_{f_i} \rightarrow O$, $i = 1, ..., k$, son funciones tales que $D_{f_i} \cap D_{f_j} = \emptyset$ para $i \neq j$, entonces $f_1 \cup ... \cup f_k$ es la funcion

$$\begin{array}{rcl} D_{f_1} \cup ... \cup D_{f_k} & \rightarrow & O \\ e & \rightarrow & \left\{ \begin{array}{lcc} f_1(e) & & \text{si } e \in D_{f_1} \\ \vdots & & \vdots \\ f_k(e) & & \text{si } e \in D_{f_k} \end{array} \right. \end{array}$$

Lemma 23 *Sean $n, m \in \omega$ y $O \in \{\omega, \Sigma^*\}$. Supongamos $f_i : D_{f_i} \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$, $i = 1, ..., k$, son funciones Σ-efectivamente computables tales que $D_{f_i} \cap D_{f_j} = \emptyset$ para $i \neq j$. Entonces $f_1 \cup ... \cup f_k$ es Σ-efectivamente computable.*

Proof. Haremos el caso $O = \Sigma^*$ y $k = 2$. Sean $\mathbb{P}_1$ y $\mathbb{P}_2$ procedimientos efectivos que computen a f_1 y f_2, respectivamente. Sea $\mathbb{P}$ el procedimiento efectivo siguiente:
- Conjunto de datos de entrada de $\mathbb{P}$ igual a $\omega^n \times \Sigma^{*m}$
- Conjunto de datos de salida de $\mathbb{P}$ contenido en Σ^*
- Funcionamiento:

Etapa 1
Hacer $T = 1$

Etapa 2
Correr el procedimiento $\mathbb{P}_1$ una cantidad T de pasos. En caso de que termine guardar la salida en la variable X e ir a Etapa 5. Si no termina ir a Etapa 3.

Etapa 3
Correr el procedimiento $\mathbb{P}_2$ una cantidad T de pasos. En caso de que termine guardar la salida en la variable X e ir a Etapa 6. Si no termina ir a Etapa 4.

Etapa 4
Hacer $T = T + 1$ e ir a Etapa 2

Etapa 5
Dar como salida el contenido de X y terminar.

Dejamos al lector corroborar que el procedimiento $\mathbb{P}$ computa a la funcion $f_1 \cup f_2$. ■

2.2 Conjuntos Σ-efectivamente computables

Un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ sera llamado Σ-*efectivamente computable* cuando la funcion $\chi_S^{\omega^n \times \Sigma^{*m}}$ sea Σ-efectivamente computable. Notese que S es Σ-efectivamente computable sii hay un procedimiento efectivo $\mathbb{P}$, el cual computa $\chi_S^{\omega^n \times \Sigma^{*m}}$, es decir:

- El conjunto de datos de entrada de $\mathbb{P}$ es $\omega^n \times \Sigma^{*m}$, siempre termina y da como dato de salida un elemento de $\{0, 1\}$.

- Dado $(\vec{x},\vec{\alpha})\in\omega^n\times\Sigma^{*m}$, $\mathbb{P}$ da como salida al numero 1 si $(\vec{x},\vec{\alpha})\in S$ y al numero 0 si $(\vec{x},\vec{\alpha})\notin S$.

Si $\mathbb{P}$ es un procedimiento efectivo el cual computa a $\chi_S^{\omega^n\times\Sigma^{*m}}$, diremos que $\mathbb{P}$ *decide la pertenecia a* S, con respecto al conjunto $\omega^n\times\Sigma^{*m}$.

Notese que esta definicion para el caso $S=\emptyset$ tiene a priori cierta ambiguedad ya que cualesquiera sean $n,m\in\omega$ tenemos que $\emptyset\subseteq\omega^n\times\Sigma^{*m}$. De todas maneras, cualesquiera sean los n,m elejidos, siempre hay un procedimiento efectivo que computa a $\chi_\emptyset^{\omega^n\times\Sigma^{*m}}$, i.e. un procedimiento que siempre da como salida 0, cualesquiera sea el dato de entrada de $\omega^n\times\Sigma^{*m}$. Es decir que el conjunto $\emptyset$ es Σ-efectivamente computable cualesquiera sea el alfabeto Σ. Cabe destacar que para el caso de un conjunto $S\neq\emptyset$, nuestra definicion es inambigua ya que hay unicos $n,m\in\omega$ tales que $S\subseteq\omega^n\times\Sigma^{*m}$.

Dejamos al lector la facil prueba del siguiente resultado.

Lemma 24 *Sean $S_1,S_2\subseteq\omega^n\times\Sigma^{*m}$ conjuntos Σ-efectivamente computables. Entonces $S_1\cup S_2, S_1\cap S_2$ y $(\omega^n\times\Sigma^{*m})-S_1$ son Σ-efectivamente computables.*

El siguiente lema caracteriza cuando un conjunto rectangular es Σ-efectivamente computable

Lemma 25 *Supongamos $S_1,...,S_n\subseteq\omega$, $L_1,...,L_m\subseteq\Sigma^*$ son conjuntos no vacios. Entonces $S_1\times...\times S_n\times L_1\times...\times L_m$ es Σ-efectivamente computable sii $S_1,...,S_n,L_1,...,L_m$ son Σ-efectivamente computables*

Proof. Notese que si $n=m=0$, entonces $S_1\times...\times S_n\times L_1\times...\times L_m=\{\Diamond\}$ el cual es Σ-efectivamente computable por lo cual el lema se cumple. Vemos entonces el caso $n+m\geq 1$. Para hacer mas lejible la prueba haremos el caso $n=m=1$. La prueba general es completamente analoga.

$(\Rightarrow)$ Ya que S_1 y L_1 son conjuntos no vacios, hay $x_0\in S_1$ y $\alpha_0\in L_1$. Ya que $S_1\times L_1$ es Σ-efectivamente computable, tenemos que $\chi_{S_1\times L_1}^{\omega\times\Sigma^*}$ es Σ-efectivamente computable. Notese que:

$$x\in S_1 \text{ sii } (x,\alpha_0)\in S_1\times L_1 \text{ sii } \chi_{S_1\times L_1}^{\omega\times\Sigma^*}(x,\alpha_0)=1$$

Por lo tanto, es facil usando un procedimiento efectivo que compute a $\chi_{S_1\times L_1}^{\omega\times\Sigma^*}$ diseñar un procedimiento efectivo que compute a $\chi_{S_1}^{\omega}$. En forma similar se prueba que L_1 es Σ-efectivamente computable.

$(\Leftarrow)$ Es facil, usando procedimientos efectivos que computen a $\chi_{S_1}^{\omega}$ y $\chi_{L_1}^{\Sigma^*}$, armar un procedimiento efectivo que compute a $\chi_{S_1\times L_1}^{\omega\times\Sigma^*}$. ■

2.3 Conjuntos Σ-efectivamente enumerables

Un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ sera llamado Σ-*efectivamente enumerable* cuando sea vacio o haya una funcion $F : \omega \rightarrow \omega^n \times \Sigma^{*m}$ tal que $I_F = S$ y $F_{(i)}$ sea Σ-efectivamente computable, para cada $i \in \{1, ..., n + m\}$. Notese que para el caso $n = m = 0$, la condicion de que $F_{(i)}$ sea Σ-efectivamente computable, para cada $i \in \{1, ..., n+m\}$ se cumple vacuamente y por lo tanto la definicion anterior nos dice que un conjunto $S \subseteq \omega^0 \times \Sigma^{*0} = \{\Diamond\}$ sera Σ-efectivamente enumerable sii es vacio o hay una funcion Σ-efectivamente computable $F : \omega \rightarrow \{\Diamond\}$, tal que $I_F = S$. Por supuesto, esto nos dice que $\emptyset$ y $\{\Diamond\}$ son Σ-efectivamente enumerables.

El siguiente resultado nos permite entender mejor la idea subyacente a esta definicion.

Lemma 26 *Un conjunto no vacio $S \subseteq \omega^n \times \Sigma^{*m}$ es Σ-efectivamente enumerable sii hay un procedimiento efectivo $\mathbb{P}$ tal que*

(1) El conjunto de datos de entrada de $\mathbb{P}$ es ω

(2) $\mathbb{P}$ se detiene para cada $x \in \omega$

(3) El conjunto de datos de salida de $\mathbb{P}$ es igual a S. (Es decir, siempre que $\mathbb{P}$ se detiene, da como salida un elemento de S, y para cada elemento $(\vec{x}, \vec{\alpha}) \in S$, hay un $x \in \omega$ tal que $\mathbb{P}$ da como salida a $(\vec{x}, \vec{\alpha})$ cuando lo corremos con x como dato de entrada)

Proof. El caso $n = m = 0$ es facil y es dejado al lector. Supongamos entonces que $n + m \geq 1$.

($\Rightarrow$) Supongamos que $S \subseteq \omega^n \times \Sigma^{*m}$ es Σ-efectivamente enumerable. Ya que S es no vacio, por definicion hay una funcion $F : \omega \rightarrow \omega^n \times \Sigma^{*m}$ tal que $I_F = S$ y cada $F_{(i)}$ es Σ-efectivamente computable. Para cada $i \in \{1, ..., n+m\}$ sea $\mathbb{P}_i$ un procedimiento efectivo que compute a $F_{(i)}$. Notar que cada $\mathbb{P}_i$ tiene a ω como conjunto de datos de entrada y siempre termina. Sea $\mathbb{P}$ el siguiente procedimiento efectivo, con conjunto de datos de entrada igual a ω y conjunto de datos de salida contenido en $\omega^n \times \Sigma^{*m}$.

Etapa 1: Correr $\mathbb{P}_1$ con dato de entrada x y alojar el dato de salida en la variable X_1

Etapa 2: Correr $\mathbb{P}_2$ con dato de entrada x y alojar el dato de salida en la variable X_2

$\vdots$

Etapa n: Correr $\mathbb{P}_n$ con dato de entrada x y alojar el dato de salida en la variable X_n

Etapa $n + 1$: Correr $\mathbb{P}_{n+1}$ con dato de entrada x y alojar el dato de salida en la variable A_1

Etapa $n + 2$: Correr $\mathbb{P}_{n+2}$ con dato de entrada x y alojar el dato de salida en la variable A_2

$\vdots$

Etapa $n+m$: Correr $\mathbb{P}_{n+m}$ con dato de entrada x y alojar el dato de salida en la variable A_m

Etapa $n+m+1$: Detenerse y dar $(X_1,...,X_n,A_1,...,A_m)$ como dato de salida

Dejamos al lector la verificacion de que el procedimiento $\mathbb{P}$ es efectivo y cumple las propiedades (1), (2) y (3).

($\Leftarrow$) Supongamos $\mathbb{P}$ es un procedimiento efectivo el cual cumple las propiedades (1), (2) y (3). Definamos $F:\omega\rightarrow\omega^n\times\Sigma^{*m}$, de la siguiente manera:

$$F(x) = \text{ dato de salida de } \mathbb{P} \text{ cuando lo corremos desde } x$$

Notar que para cada $i\in\{1,...,n+m\}$ la funcion $F_{(i)}$ es Σ-efectivamente computable ya que el siguiente procedimiento efectivo la computa:

Etapa 1: Correr $\mathbb{P}$ desde x y guardar la salida en la variable V

Etapa 2: Dar como salida la coordenada i-esima de V ■

Cuando un procedimiento $\mathbb{P}$ cumpla (1), (2) y (3) del lema anterior, diremos que $\mathbb{P}$ *enumera* a S. O sea que S es Σ-efectivamente enumerable sii es vacio o hay un procedimiento efectivo el cual enumera a S.

Dicho de otra forma un conjunto no vacio S es Σ-efectivamente enumerable sii hay un procedimiento efectivo $\mathbb{P}$ el cual tiene conjunto de datos de entrada ω y ademas para los sucesivos datos de entrada $0,1,2,3,...$, el procedimiento $\mathbb{P}$ produce respectivamente los datos de salida $e_0,e_1,e_2,e_3,...$ de manera que $S=\{e_0,e_1,e_2,...\}$. Cabe destacar aqui que puede suceder que $e_i=e_j$, para ciertos i,j, con $i\neq j$.

Algunos ejemplos:

E$_1$ El conjunto $S=\{x\in\omega : x \text{ es par}\}$ es Σ-efectivamente enumerable, cualesquiera sea Σ. El siguiente procedimiento enumera a S:

- Calcular $2x$, darlo como dato de salida y terminar.

E$_2$ Un procedimiento que enumera $\omega\times\omega$ es el siguiente:

Etapa 1

Si $x=0$, dar como salida el par $(0,0)$ y terminar. Si $x\neq 0$, calcular $x_1=(x)_1$ y $x_2=(x)_2$.

Etapa 2

Dar como dato de salida el par (x_1,x_2) y terminar

Como puede notarse el procedimiento es efectivo y ademas el conjunto de datos de salida es $\omega\times\omega$ ya que si tomamos un par cualquiera $(a,b)\in\omega\times\omega$, el procedimiento lo dara como dato de salida para la entrada $x=2^a3^b$.

E$_3$ Veamos que $\omega^2\times\Sigma^{*3}$ es Σ-efectivamente enumerable cualquiera sea el alfabeto no vacio Σ. Sea $\leq$ un orden total para el alfabeto Σ. Utilisando el orden $<$ podemos diseñar el siguiente procedimiento para enumerar $\omega^2\times\Sigma^{*3}$:

Etapa 1

Si $x = 0$, dar como salida $(0, 0, \varepsilon, \varepsilon, \varepsilon)$ y terminar. Si $x \neq 0$, calcular

$x_1 = (x)_1$

$x_2 = (x)_2$

$\alpha_1 = *^{\leq}((x)_3)$

$\alpha_2 = *^{\leq}((x)_4)$

$\alpha_3 = *^{\leq}((x)_5)$

Etapa 2

Dar como dato de salida la 5-upla $(x_1, x_2, \alpha_1, \alpha_2, \alpha_3)$.

Lemma 27 *Sean $S_1, S_2 \subseteq \omega^n \times \Sigma^{*m}$ conjuntos Σ-efectivamente enumerables. Entonces $S_1 \cup S_2$ y $S_1 \cap S_2$ son Σ-efectivamente enumerables.*

Proof. El caso en el que alguno de los conjuntos es vacio es trivial. Supongamos que ambos conjuntos son no vacios y sean $\mathbb{P}_1$ y $\mathbb{P}_2$ procedimientos que enumeran a S_1 y S_2. El siguiente procedimiento enumera al conjunto $S_1 \cup S_2$:

- Si x es par realizar $\mathbb{P}_1$ partiendo de $x/2$ y dar el elemento de S_1 obtenido como salida. Si x es impar realizar $\mathbb{P}_2$ partiendo de $(x-1)/2$ y dar el elemento de S_2 obtenido como salida.

Veamos ahora que $S_1 \cap S_2$ es Σ-efectivamente enumerable. Si $S_1 \cap S_2 = \emptyset$ entonces no hay nada que probar. Supongamos entonces que $S_1 \cap S_2$ es no vacio. Sea e_0 un elemento fijo de $S_1 \cap S_2$. Sea $\mathbb{P}$ un procedimiento efectivo el cual enumere a $\omega \times \omega$ (ver el ejemplo de mas arriba). Un procedimiento que enumera a $S_1 \cap S_2$ es el siguiente

Etapa 1

Realizar $\mathbb{P}$ con dato de entrada x, para obtener un par $(x_1, x_2) \in \omega \times \omega$.

Etapa 2

Realizar $\mathbb{P}_1$ con dato de entrada x_1 para obtener un elemento $e_1 \in S_1$

Etapa 3

Realizar $\mathbb{P}_2$ con dato de entrada x_2 para obtener un elemento $e_2 \in S_2$

Etapa 4

Si $e_1 = e_2$, entonces dar como dato de salida e_1. En caso contrario dar como dato de salida e_0.

Dejamos al lector la prueba de que este procedimiento enumera a $S_1 \cap S_2$.

■

Dejamos al lector la prueba del siguiente resultado.

Lemma 28 *Supongamos $S_1, ..., S_n \subseteq \omega$, $L_1, ..., L_m \subseteq \Sigma^*$ son conjuntos no vacios. Entonces $S_1 \times ... \times S_n \times L_1 \times ... \times L_m$ es Σ-efectivamente enumerable sii $S_1, ..., S_n, L_1, ..., L_m$ son Σ-efectivamente enumerables*

@@finpagina@@

Lemma 29 *Si* $S \subseteq \omega^n \times \Sigma^{*m}$ *es* Σ*-efectivamente computable entonces* S *es* Σ*-efectivamente enumerable.*

Proof. Supongamos $S \neq \emptyset$. Sea $(\vec{z}, \gamma) \in S$, fijo. Sea $\mathbb{P}$ un procedimiento efectivo que compute a $\chi_S^{\omega^n \times \Sigma^{*m}}$. Ya vimos en el ejemplo anterior que $\omega^2 \times \Sigma^{*3}$ es Σ-efectivamente enumerable. En forma similar se puede ver que $\omega^n \times \Sigma^{*m}$ lo es. Sea $\mathbb{P}_1$ un procedimiento efectivo que enumere a $\omega^n \times \Sigma^{*m}$. Entonces el siguiente procedimiento enumera a S:

Etapa 1

Realizar $\mathbb{P}_1$ con x de entrada para obtener $(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m}$.

Etapa 2

Realizar $\mathbb{P}$ con $(\vec{x}, \vec{\alpha})$ de entrada para obtener el valor Booleano e de salida.

Etapa 3

Si $e = 1$ dar como dato de salida $(\vec{x}, \vec{\alpha})$. Si $e = 0$ dar como dato de salida $(\vec{z}, \gamma)$.

■

Como veremos mas adelante en la materia (Proposicion 136), hay conjuntos que son Σ-efectivamente enumerables y no Σ-efectivamente computables. Sin envargo tenemos el siguiente interesante resultado:

Theorem 30 *Sea* $S \subseteq \omega^n \times \Sigma^{*m}$*. Son equivalentes*

(a) S *es* Σ*-efectivamente computable*

(b) S *y* $(\omega^n \times \Sigma^{*m}) - S$ *son* Σ*-efectivamente enumerables*

Proof. (a)⇒(b). Por el lema anterior tenemos que S es Σ-efectivamente enumerable. Notese ademas que, dado que S es Σ-efectivamente computable, $(\omega^n \times \Sigma^{*m}) - S$ tambien lo es (por que?). Es decir que aplicando nuevamente el lema anterior tenemos que $(\omega^n \times \Sigma^{*m}) - S$ es Σ-efectivamente enumerable.

(b)⇒(a). Si $S = \emptyset$ o $S = \omega^n \times \Sigma^{*m}$ es claro que se cumple (a). O sea que podemos suponer que ni S ni $\omega^n \times \Sigma^{*m}$ son igual al conjunto vacio. Sea $\mathbb{P}_1$ un procedimiento efectivo que enumere a S y sea $\mathbb{P}_2$ un procedimiento efectivo que enumere a $(\omega^n \times \Sigma^{*m}) - S$. Es facil ver que el siguiente procedimiento computa el predicado $\chi_S^{\omega^n \times \Sigma^{*m}}$:

Etapa 1

Darle a la variable T el valor 0.

Etapa 2

Realizar $\mathbb{P}_1$ con el valor de T como entrada para obtener de salida la upla $(\vec{y}, \vec{\beta})$.

Etapa 3

Realizar $\mathbb{P}_2$ con el valor de T como entrada para obtener de salida la upla $(\vec{z}, \vec{\gamma})$.

Etapa 4

Si $(\vec{y}, \vec{\beta}) = (\vec{x}, \vec{\alpha})$, entonces detenerse y dar como dato de salida el valor 1. Si $(\vec{z}, \vec{\gamma}) = (\vec{x}, \vec{\alpha})$, entonces detenerse y dar como dato de salida el valor 0. Si no suceden ninguna de las dos posibilidades antes mensionadas, aumentar en 1 el valor de la variable T y dirijirse a la Etapa 2. ■

Supongamos que $k, l, n, m \in \omega$ y que $F : D_F \subseteq \omega^k \times \Sigma^{*l} \to \omega^n \times \Sigma^{*m}$. Supongamos ademas que $n + m \geq 1$. Entonces denotaremos con $F_{(i)}$ a la funcion $p_i^{n,m} \circ F$. Notar que

$$\begin{array}{lll} F_{(i)} & : & D_F \subseteq \omega^k \times \Sigma^{*l} \to \omega \text{, para cada } i = 1, ..., n \\ F_{(i)} & : & D_F \subseteq \omega^k \times \Sigma^{*l} \to \Sigma^* \text{, para cada } i = n+1, ..., n+m \end{array}$$

Por lo cual cada una de las funciones $F_{(i)}$ son Σ-mixtas. Ademas notese que

$$F = [F_{(1)}, ..., F_{(n+m)}]$$

Theorem 31 *Dado $S \subseteq \omega^n \times \Sigma^{*m}$, son equivalentes*

(1) S es Σ-efectivamente enumerable

(2) $S = I_F$, para alguna $F : D_F \subseteq \omega^k \times \Sigma^{*l} \to \omega^n \times \Sigma^{*m}$ tal que cada $F_{(i)}$ es Σ-efectivamente computable.

(3) $S = D_f$, para alguna funcion f la cual es Σ-efectivamente computable.

Proof. El caso $n = m = 0$ es facil y es dejado al lector. Supongamos entonces que $n + m \geq 1$.

(1)⇒(2) es trivial.

(2)⇒(3). Para $i = 1, ..., n + m$, sea $\mathbb{P}_i$ un procedimiento el cual computa a $F_{(i)}$ y sea $\mathbb{P}$ un procedimiento el cual enumere a $\omega \times \omega^k \times \Sigma^{*l}$. El siguiente procedimiento computa la funcion $f : I_F \to \{1\}$:

Etapa 1

Darle a la variable T el valor 0.

Etapa 2

Hacer correr $\mathbb{P}$ con dato de entrada T y obtener $(t, z_1, ..., z_k, \gamma_1, ..., \gamma_l)$ como dato de salida.

Etapa 3

Para cada $i = 1, ..., n+m$, hacer correr $\mathbb{P}_i$ durante t pasos, con dato de entrada $(z_1, ..., z_k, \gamma_1, ..., \gamma_l)$. Si cada procedimiento $\mathbb{P}_i$ al cabo de los t pasos termino y dio como resultado el valor o_i, entonces comparar $(\vec{x}, \vec{\alpha})$ con $(o_1, ..., o_{n+m})$ y en caso de que sean iguales detenerse y dar como dato de salida el valor 1. En el caso en que no son iguales, aumentar en 1 el valor de la variable T y dirijirse a la Etapa 2. Si algun procedimiento $\mathbb{P}_i$ al cabo de los t pasos no termino, entonces aumentar en 1 el valor de la variable T y dirijirse a la Etapa 2.

(3)⇒(1). Supongamos $S \neq \emptyset$. Sea $(\vec{z}, \vec{\gamma})$ un elemento fijo de S. Sea $\mathbb{P}$ un procedimiento el cual compute a f. Sea $\mathbb{P}_1$ un procedimiento el cual enumere a $\omega \times \omega^n \times \Sigma^{*m}$. Dejamos al lector el diseño de un procedimiento efectivo el cual enumere D_f. ■

Dejamos como ejercicio la prueba de los dos siguientes lemas.

Lemma 32 *Sean $n, m \in \omega$ y $O \in \{\omega, \Sigma^*\}$. Supongamos $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to O$ es Σ-efectivamente computable y $S \subseteq I_f$ es Σ-efectivamente enumerable, entonces $f^{-1}(S) = \{(\vec{x}, \vec{\alpha}) : f(\vec{x}, \vec{\alpha}) \in S\}$ es Σ-efectivamente enumerable*

Lemma 33 *Sean $n, m \in \omega$ y $O \in \{\omega, \Sigma^*\}$. Supongamos $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to O$ es Σ-efectivamente computable y $S \subseteq D_f$ es Σ-efectivamente enumerable, entonces $f|_S$ es Σ-efectivamente computable*

2.4 Independencia del alfabeto

Una observacion importante es que los conceptos de funcion Σ-efectivamente computable, de conjunto Σ-efectvamente computable y de conjunto Σ-efectivamente enumerable, no dependen del alfabeto Σ. Esto lo establecemos formalmente en los dos siguientes lemas.

Lemma 34 *Sean Σ y Γ alfabetos cualesquiera. Supongamos una funcion f es Σ-mixta y Γ-mixta, entonces f es Σ-efectivamente computable sii f es Γ-efectivamente computable.*

Lemma 35 *Sean Σ y Γ alfabetos cualesquiera. Supongamos un conjunto S es Σ-mixto y Γ-mixto, entonces S es Σ-efectivamente computable (resp. Σ-efectivamente enumerable) sii S es Γ-efectivamente computable (resp. Γ-efectivamente enumerable).*

Dejamos al lector los detalles de las rutinarias pruebas de estos dos lemas.

3 Tres modelos matematicos de la computabilidad efectiva

Ya que el concepto de procedimiento efectivo es un concepto intuitivo, impresiso y a priori no expresado en el formalismo matematico, los conceptos de

- Funcion Σ-efectivamente computable
- Conjunto Σ-efectivamente computable
- Conjunto Σ-efectivamente enumerable

tambien son impresisos y estan fuera del formalismo matematico, debido a que los tres se definen en terminos de la existencia de procedimientos efectivos. Por supuesto, los tres conceptos son fundamentales en el estudio teorico de la computabilidad por lo que es muy importante poder dar un modelo o formalizacion matematica de estos conceptos. Pero notese que los dos ultimos se definen en funcion del primero por lo que una formalizacion matematica precisa del concepto de funcion Σ-efectivamente computable, resuelve el problema de modelizar en forma matematica estos a tres conceptos.

En esta seccion daremos las tres formalizaciones matematicas mas clasicas del concepto de funcion Σ-efectivamente computable. La primera y la mas apegada a la idea intuitiva de procedimiento efectivo es la dada por Alan Turing via la matematizacion del concepto de maquina. La segunda, es la dada por Godel en su estudio de sistemas formales de la logica de primer orden. Por ultimo veremos la formalizacion via un lenguaje de programacion imperativo. En honor a la influencia que tuvo Von Neumann en el diseño de la primer computadora de caracter universal (i.e. programable de proposito general), llamaremos a este paradigma el paradigma imperativo de Von Neumann.

3.1 El paradigma de Turing

Estudiaremos el concepto de maquina de Turing, el cual fue introducido por Alam Turing para formalizar o modelizar matematicamente la idea de procedimiento efectivo. Una vez definidas las maquinas podremos dar una modelizacion matematica precisa del concepto de funcion Σ-efectivamente computable. Llamaremos a estas funciones Σ-Turing computables y seran aquellas que (en algun sentido que sera bien precisado matematicamente) pueden ser computadas por una maquina de Turing. Por supuesto, la fidedignidad de este concepto, es decir cuan buena es la modelizacion matematica dada por Turing, puede no ser clara al comienzo pero a medida que vayamos avanzando en nuestro estudio y conozcamos ademas los otros paradigmas y su relacion, quedara claro que el modelo de Turing es acertado.

Vivimos en un mundo plagado de maquinas (ascensores, celulares, relojes, taladros, etc). Una caracteristica comun a todas las maquinas es que tienen distintos estados posibles. Un estado es el conjunto de caracteristicas que determinan un momento concreto posible de la maquina cuando esta funcionando. Por ejemplo un estado posible de un ascensor seria:

- esta en el tercer piso, con la primer puerta abierta y la otra cerrada, esta apretado el boton de ir al sexto piso, etc

donde ponemos etc porque dependiendo del tipo de ascensor (si es con memoria, a que pisos puede ir, etc) habra mas datos que especificar para determinar un estado concreto.

Otra caracteristica comun de las maquinas es que interactuan de distintas formas con el usuario o mas generalmente su entorno. Dependiendo de que accion se ejecute sobre la maquina y en que estado este, la maquina realizara alguna tarea y ademas cambiara de estado. En general las maquinas son *deterministicas* en el sentido que siempre que esten en determinado estado y se les aplique determinada accion, realizaran la misma tarea y pasaran al mismo estado.

3.1.1 Descripcion informal de las maquinas de Turing

Son un modelo abstracto de maquina con una cantidad finita de estados la cual trabaja sobre una cinta de papel dividida en cuadros e interactua o recibe acciones externas por medio de una cabeza lectora que lee de a un cuadro de la cinta a la ves y ademas puede borrar el contenido del cuadro leido y escribir en el un simbolo. Tambien la cabeza lectora puede moverse un cuadro hacia la izquierda o hacia la derecha. La cinta tiene un primer cuadro hacia su izquierda pero hacia la derecha puede extenderse todo lo necesario. En un cuadro de la cinta podra haber un simbolo o un cuadro puede simplemente estar en blanco. Es decir que habra un alfabeto Γ el cual consiste de todos los simbolos que pueden figurar en la cinta. Esto sera parte de los datos o caracteristicas de cada maquina, es decir, Γ puede cambiar dependiendo de la maquina. La maquina, en funcion del estado en que se encuentre y de lo que vea su cabeza lectora en el cuadro escaneado, podra moverse a lo sumo un cuadro (izquierda, derecha o quedarse quieta), modificar lo que encuentre en dicho cuadro (borrando y escribiendo algun nuevo simbolo) y cambiar de estado (posiblemente al mismo que tenia). Para simplificar supondremos que hay en Γ un simbolo el cual si aparece en un cuadro de la cinta, significara que dicho cuadro esta sin escribir o en blanco. Esto nos permitira describir mas facilmente el funcionamiento de la maquina. En gral llamaremos B a tal simbolo. Tambien por lo general llamaremos Q al conjunto de estados de la maquina.

Tambien cada maquina tendra un estado especial el cual sera llamado su estado inicial, generalmente denotado con q_0, el cual sera el estado en el que estara la maquina al comenzar a trabajar sobre la cinta. Hay otras caracteristicas que tendran las maquinas de Turing pero para dar un primer ejemplo ya nos basta. Describiremos una maquina de Turing M que tendra $\Gamma = \{@, a, b, B\}$ y tendra dos estados, es decir $Q = \{q_0, q_1\}$. Obviamente q_0 sera su estado inicial y ademas el "comportamiento o personalidad" de M estara dado por las siguientes clausulas:

- Estando en estado q_0 si ve ya sea b o B o @, entonces se queda en estado q_0 y se mueve a la derecha

- Estando en estado q_0 si ve a entonces reescribe @, se mueve a la izquierda y cambia al estado q_1

- Estando en estado q_1 si ve a o b o B o @ entonces lo deja como esta, se mueve a la izquierda y queda en estado q_1

Supongamos ahora que tomamos una palabra $\alpha \in \Gamma^*$ y la distribuimos en la cinta dejando el primer cuadro en blanco y luego poniendo los simbolos de α en los siguientes cuadros. Supongamos ademas que ponemos la maquina en estado q_0 y con su cabeza lectora escaneando el primer cuadro de la cinta. Esto lo podemos representar graficamente de la siguiente manera

$$\begin{array}{llllllll} B & \alpha_1 & ... & \alpha_n & B & B & B & ... \\ \uparrow \\ q_0 \end{array}$$

donde $\alpha_1, ..., \alpha_n$ son los sucesivos simbolos de α. Supongamos ademas que a ocurre an α. Dejamos al lector ir aplicando las clausulas de M para convencerse que luego de un rato de funcionar M, la situacion sera

$$\begin{array}{llllllll} B & \beta_1 & ... & \beta_n & B & B & B & ... \\ \uparrow \\ q_1 \end{array}$$

donde $\beta_1 ... \beta_n$ es el resultado de reemplazar en α la primer ocurrencia de a por @. Dejamos como ejercicio para el lector averiguar que sucede cuando a no ocurre en α

Una cosa que puede pasar es que para un determinado estado p y un $\sigma \in \Gamma$, la maquina no tenga contemplada ninguna accion posible. Por ejemplo sea M la maquina de Turing dada por $Q = \{q_0\}$, $\Gamma = \{@, \$, B\}$ y por la siguiente clausula:

- Estando en estado q_0 si ve ya sea @ o B, entonces se queda en estado q_0 y se mueve a la derecha

Es facil ver que si partimos de una situacion

$$\begin{array}{llllllll} B & \alpha_1 & ... & \alpha_n & B & B & B & ... \\ \uparrow \\ q_0 \end{array}$$

donde $\alpha_1, ..., \alpha_n \in \Gamma$, entonces si ningun α_i es igual a \$, la maquina se movera indefinidamente hacia la derecha y en caso contrario se movera i pasos a la derecha y se detendra, donde i es el menor l tal que $\alpha_l = \$$.

Otro caso posible de detencion de una maquina de Turing es cuando esta escaneando el primer cuadro de la cinta y su unica accion posible implica moverse un cuadro a la izquierda. Tambien en estos casos diremos que la maquina se detiene ya que la cinta no es extensible hacia la izquierda.

Otra caracteristica de las maquinas de Turing es que poseen un *alfabeto de entrada* el cual esta contenido en el alfabeto Γ y en el cual estan los simbolos que se usaran para formar la configuracion inicial de la cinta (exepto B). En general lo denotaremos con Σ al alfabeto de entrada y los simbolos de $\Gamma - \Sigma$ son considerados auxiliares. Tambien habra un conjunto F contenido en el conjunto Q de los estados de la maquina, cuyos elementos seran llamados *estados finales*.

Diremos que una palabra $\alpha = \alpha_1...\alpha_n \in \Sigma^*$ es *aceptada por M por alcance de estado final* si partiendo de

$$\begin{array}{llllllll} B & \alpha_1 & ... & \alpha_n & B & B & B & ... \\ \uparrow & & & & & & & \\ q_0 & & & & & & & \end{array}$$

en algun momento de la computacion M esta en un estado de F. Llamaremos $L(M)$ al conjunto formado por todas las palabras que son aceptadas por alcance de estado final

Diremos que una palabra $\alpha = \alpha_1...\alpha_n \in \Sigma^*$ es *aceptada por M por detencion* si partiendo de

$$\begin{array}{llllllll} B & \alpha_1 & ... & \alpha_n & B & B & B & ... \\ \uparrow & & & & & & & \\ q_0 & & & & & & & \end{array}$$

en algun momento M se detiene. Llamaremos $H(M)$ al conjunto formado por todas las palabras que son aceptadas por detencion

3.1.2 Definicion matematica de maquina de Turing

Una *maquina de Turing* es una 7-upla $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ donde

- Q es un conjunto finito cuyos elementos son llamados *estados*
- Γ es un alfabeto que contiene a Σ
- Σ es un alfabeto llamado el *alfabeto de entrada*
- $B \in \Gamma - \Sigma$ es un simbolo de Γ llamado el *blank symbol*
- $\delta : D_\delta \subseteq Q \times \Gamma \to Q \times \Gamma \times \{L, R, K\}$
- q_0 es un estado llamado el *estado inicial* de M
- $F \subseteq Q$ es un conjunto de estados llamados *finales*

Notese que la funcion δ da la "personalidad" de la maquina. Aqui los simbolos L, R, K serviran para especificar que hace el cabezal. O sea:

- $\delta(p, \sigma) = (q, \gamma, L)$ significara que la maquina estando en estado p y leyendo el simbolo σ borrara σ y escribira γ en su lugar y luego se movera un cuadro a la izquierda (esto en caso que el cabezal no este en el cuadro de mas a la izquierda, en cuyo caso no podra realizar dicha tarea y se detendra).

- $\delta(p,\sigma) = (q,\gamma,K)$ significara que la maquina estando en estado p y leyendo el simbolo σ borrara σ y escribira γ en su lugar y luego el cabezal se quedara kieto

- $\delta(p,\sigma) = (q,\gamma,R)$ significara que la maquina estando en estado p y leyendo el simbolo σ borrara σ y escribira γ en su lugar y luego el cabezal se movera un cuadro a la derecha

Si bien en nuestra definicion de maquina de Turing no hay ninguna restriccion acerca de la naturaleza de los elementos de Q, para continuar nuestro analisis asumiremos siempre que Q es un alfabeto disjunto con Γ. Esto nos permitira dar definiciones matematicas precisas que formalizaran el funcionamiento de las maquinas de Turing en el contexto de las funciones mixtas. Deberia quedar claro que el hecho que solo analicemos maquinas en las cuales Q es un alfabeto disjunto con Γ, no afectara la profundidad y generalidad de nuestros resultados.

Descripciones instantaneas Una *descripcion instantanea* sera una palabra de la forma $\alpha q\beta$, donde $\alpha,\beta \in \Gamma^*$, $[\beta]_{|\beta|} \neq B$ y $q \in Q$. La descripcion instantanea $\alpha_1...\alpha_n q\beta_1...\beta_m$, con $\alpha_1,...,\alpha_n,\ \beta_1,...,\beta_m \in \Gamma,\ n,m \geq 0$ representara la siguiente situacion

$$\begin{array}{cccccccccccc} \alpha_1 & \alpha_2 & ... & \alpha_n & \beta_1 & \beta_2 & ... & \beta_m & B & B & B & ... \\ & & & & \uparrow & & & & & & & \\ & & & & q & & & & & & & \end{array}$$

Usaremos Des para denotar el conjunto de las descripciones instantaneas. Definamos la funcion $St : Des \to Q$, de la siguiente manera

$$St(d) = \text{unico simbolo de } Q \text{ que ocurre en } d$$

La relacion $\vdash$ Dado $\alpha \in (Q \cup \Gamma)^*$, definamos $\lfloor\alpha\rfloor$ de la siguiente manera

$$\begin{array}{rcl} \lfloor\varepsilon\rfloor & = & \varepsilon \\ \lfloor\alpha\sigma\rfloor & = & \alpha\sigma, \text{ si } \sigma \neq B \\ \lfloor\alpha B\rfloor & = & \lfloor\alpha\rfloor \end{array}$$

Es decir $\lfloor\alpha\rfloor$ es el resultado de remover de α el tramo final mas grande de la forma B^n. Dada cualquier palabra α definimos

$$^{\frown}\alpha = \left\{ \begin{array}{lll} [\alpha]_2 \dots [\alpha]_{|\alpha|} & \text{si} & |\alpha| \geq 2 \\ \varepsilon & \text{si} & |\alpha| \leq 1 \end{array} \right. \qquad \alpha^{\frown} = \left\{ \begin{array}{lll} [\alpha]_1 \dots [\alpha]_{|\alpha|-1} & \text{si} & |\alpha| \geq 2 \\ \varepsilon & \text{si} & |\alpha| \leq 1 \end{array} \right.$$

Dadas $d_1, d_2 \in Des$, escribiremos $d_1 \vdash d_2$ cuando existan $\sigma \in \Gamma$, $\alpha,\beta \in \Gamma^*$ y $p,q \in Q$ tales que se cumple alguno de los siguientes casos

Caso 1.

$$\begin{array}{rcl} d_1 & = & \alpha p\beta \\ \delta\left(p,[\beta B]_1\right) & = & (q,\sigma,R) \\ d_2 & = & \alpha\sigma q^\frown\beta \end{array}$$

Caso 2.

$$\begin{array}{rcl} d_1 & = & \alpha p\beta \\ \delta\left(p,[\beta B]_1\right) & = & (q,\sigma,L) \text{ y } \alpha\neq\varepsilon \\ d_2 & = & \left\lfloor \alpha^\frown q\,[\alpha]_{|\alpha|}\,\sigma^\frown\beta\right\rfloor \end{array}$$

Caso 3.

$$\begin{array}{rcl} d_1 & = & \alpha p\beta \\ \delta(p,[\beta B]_1) & = & (q,\sigma,K) \\ d_2 & = & \lfloor \alpha q\sigma^\frown\beta\rfloor \end{array}$$

Escribiremos $d\nvdash d'$ para expresar que no se da $d\vdash d'$. Para $d,d'\in Des$ y $n\geq 0$, escribiremos $d\overset{n}{\vdash} d'$ si existen $d_1,...,d_{n+1}\in Des$ tales que

$$\begin{array}{rcl} d & = & d_1 \\ d' & = & d_{n+1} \\ d_i & \vdash & d_{i+1}, \text{ para } i=1,...,n. \end{array}$$

Notese que $d\overset{0}{\vdash} d'$ sii $d=d'$. Finalmente definamos

$$d\overset{*}{\vdash} d' \text{ sii } (\exists n\in\omega)\ d\overset{n}{\vdash} d'.$$

Detencion Dada $d\in Des$, diremos que M *se detiene partiendo de* d si existe $d'\in Des$ tal que

- $d\overset{*}{\vdash} d'$
- $d'\nvdash d''$, para cada $d''\in Des$

Deberia quedar claro que es posible que $\alpha p\beta\nvdash d$, para cada descripcion instantanea d, y que $\delta(p,[\beta B]_1)$ sea no vacio.

El lenguaje $L(M)$ Diremos que una palabra $\alpha\in\Sigma^*$ es *aceptada por* M *por alcance de estado final* cuando

$$\lfloor q_0B\alpha\rfloor \overset{*}{\vdash} d, \text{ con } d \text{ tal que } St(d)\in F.$$

El *lenguage aceptado por* M *por alcance de estado final* se define de la siguiente manera

$$L(M)=\{\alpha\in\Sigma^* : \alpha \text{ es aceptada por } M \text{ por alcance de estado final}\}.$$

El lenguaje $H(M)$ Diremos que una palabra $\alpha \in \Sigma^*$ es *aceptada por* M *por detencion* cuando M se detiene partiendo de $\lfloor q_0 B\alpha \rfloor$. El *lenguage aceptado por* M *por detencion* se define de la siguiente manera

$$H(M) = \{\alpha \in \Sigma^* : \alpha \text{ es aceptada por } M \text{ por detencion}\}$$

3.1.3 Funciones Σ-Turing computables

Para poder computar funciones mixtas con una maquina de Turing necesitaremos un simbolo para representar numeros sobre la cinta. Llamaremos a este simbolo *unit* y lo denotaremos con $\shortmid$. Mas formalmente una *maquina de Turing con unit* es una 8-upla $M = (Q, \Sigma, \Gamma, \delta, q_0, B, \shortmid, F)$ tal que $(Q, \Sigma, \Gamma, \delta, q_0, B, F)$ es una maquina de Turing y $\shortmid$ es un simbolo distingido perteneciente a $\Gamma - (\{B\} \cup \Sigma)$.

Diremos que una funcion $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to \Sigma^*$ es Σ-*Turing computable* si existe una maquina de Turing con unit, $M = (Q, \Sigma, \Gamma, \delta, q_0, B, \shortmid, F)$ tal que:

(1) Si $(\vec{x}, \vec{\alpha}) \in D_f$, entonces hay un $p \in Q$ tal que

$$\lfloor q_0 B \shortmid^{x_1} B...B \shortmid^{x_n} B\alpha_1 B...B\alpha_m \rfloor \overset{*}{\vdash} \lfloor pBf(\vec{x}, \vec{\alpha}) \rfloor$$

y $\lfloor pBf(\vec{x}, \vec{\alpha}) \rfloor \not\vdash d$, para cada $d \in Des$

(2) Si $(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m} - D_f$, entonces M no se detiene partiendo de

$$\lfloor q_0 B \shortmid^{x_1} B...B \shortmid^{x_n} B\alpha_1 B...B\alpha_m \rfloor .$$

En forma similar, una funcion $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to \omega$, es llamada Σ-*Turing computable* si existe una maquina de Turing con unit, $M = (Q, \Sigma, \Gamma, \delta, q_0, B, \shortmid, F)$, tal que:

(1) Si $(\vec{x}, \vec{\alpha}) \in D_f$, entonces hay un $p \in Q$ tal que

$$\lfloor q_0 B \shortmid^{x_1} B...B \shortmid^{x_n} B\alpha_1 B...B\alpha_m \rfloor \overset{*}{\vdash} \left\lfloor pB \shortmid^{f(\vec{x}, \vec{\alpha})} \right\rfloor$$

y $\left\lfloor pB \shortmid^{f(\vec{x}, \vec{\alpha})} \right\rfloor \not\vdash d$, para cada $d \in Des$

(2) Si $(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m} - D_f$, entonces M no se detiene partiendo de

$$\lfloor q_0 B \shortmid^{x_1} B...B \shortmid^{x_n} B\alpha_1 B...B\alpha_m \rfloor$$

Cuando M y f cumplan los items (1) y (2) de la definicion anterior, diremos que la funcion f es *computada* por M.

Por supuesto esta definicion no tendria sentido como modelo matematico del concepto de funcion Σ-efectivamente computable si no sucediera que toda funcion Σ-Turing computable fuera Σ-efectivamente computable. Este hecho es intuitivamente claro y lo expresamos en forma de proposicion.

Proposition 36 *Sean $n, m \in \omega$ y $O \in \{\omega, \Sigma^*\}$. Si $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to O$ es computada por una maquina de Turing deterministica con unit $M = (Q, \Sigma, \Gamma, \delta, q_0, B, \shortmid, F)$, entonces f es Σ-efectivamente computable.*

Proof. Haremos el caso $O = \Sigma^*$. Sea $\mathbb{P}$ el siguiente procedimiento efectivo.

- Conjunto de datos de entrada de $\mathbb{P}$ igual a $\omega^n \times \Sigma^{*m}$
- Conjunto de datos de salida de $\mathbb{P}$ contenido en O
- Funcionamiento: Hacer funcionar paso a paso la maquina M partiendo de la descripcion instantanea $\lfloor q_0 B \shortmid^{x_1} B...B \shortmid^{x_n} B\alpha_1 B...B\alpha_m \rfloor$. Si en alguna instancia M termina, dar como salida el resultado de remover de la descripcion instantanea final los dos primeros simbolos.

Notese que este procedimiento termina solo en aquelos elementos $(\vec{x}, \vec{\sigma}) \in \omega^n \times \Sigma^{*m}$ tales que la maquina M termina partiendo desde

$$\lfloor q_0 B \shortmid^{x_1} B...B \shortmid^{x_n} B\alpha_1 B...B\alpha_m \rfloor$$

por lo cual termina solo en los elementos de D_f ya que M computa a f. Ademas es claro que en caso de terminacion el procedimiento da como salida $f(\vec{x}, \vec{\sigma})$. ∎

Sin envargo el modelo Turingniano podria a priori no ser del todo correcto ya que podria pasar que haya una funcion que sea computada por un procedimiento efectivo pero que no exista una maquina de Turing que la compute. En otras palabras el modelo podria ser incompleto. La completitud de este modelo puede no ser clara al comienzo pero a medida que vayamos avanzando en nuestro estudio y conozcamos ademas los otros paradigmas y su relacion, quedara claro que el modelo de Turing es acertado.

3.1.4 Conjuntos Σ-Turing enumerables

Ya que la nocion de funcion Σ-Turing computable es el modelo matematico de Turing del concepto de funcion Σ-efectivamente computable, nos podriamos preguntar entonces cual es el modelo matematico de Turing del concepto de conjunto Σ-efectivamente enumerable. Si prestamos atencion a la definicion de conjunto Σ-efectivamente enumerable, notaremos que depende de la existencia de ciertas funciones Σ-efectivamente computables por lo cual la siguiente definicion cae de maduro:

Diremos que un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ sera llamado Σ-*Turing enumerable* cuando sea vacio o haya una funcion $F : \omega \to \omega^n \times \Sigma^{*m}$ tal que $I_F = S$ y $F_{(i)}$ sea Σ-Turing computable, para cada $i \in \{1, ..., n + m\}$.

Deberia quedar claro que si el concepto de funcion Σ-Turing computable modeliza correctamente al concepto de funcion Σ-efectivamente computable, entonces el concepto de conjunto Σ-Turing enumerable recien definido modeliza correctamente al concepto de conjunto Σ-efectivamente enumerable. Notese que segun la definicion que acabamos de escribir, un conjunto no vacio $S \subseteq \omega^n \times \Sigma^{*m}$ es Σ-Turing enumerable si y solo si hay maquinas de Turing deterministicas con unit

$$\begin{aligned}
M_1 &= (Q_1, \Sigma, \Gamma_1, \delta_1, q_{01}, B, \shortmid, F_1) \\
M_2 &= (Q_2, \Sigma, \Gamma_2, \delta_2, q_{02}, B, \shortmid, F_2) \\
&\vdots \\
M_{n+m} &= (Q_{n+m}, \Sigma, \Gamma_{n+m}, \delta_{n+m}, q_{0n+m}, B, \shortmid, F_{n+m})
\end{aligned}$$

tales que

- cada M_i, con $i = 1, ..., n$, computa una funcion $F_i : \omega \to \omega$
- cada M_i, con $i = n+1, ..., n+m$, computa una funcion $F_i : \omega \to \Sigma^*$
- $S = \mathrm{Im}[F_1, ..., F_{n+m}]$

Como puede notarse las maquinas $M_1, ..., M_{n+m}$ puestas en paralelo a funcionar desde la descripciones instantaneas

$$\begin{gathered}
\lfloor q_{01} B \shortmid^x \rfloor \\
\lfloor q_{02} B \shortmid^x \rfloor \\
\vdots \\
\lfloor q_{0n+m} B \shortmid^x \rfloor
\end{gathered}$$

producen en forma natural un procedimiento efectivo (con dato de entrada $x \in \omega$) que enumera a S Por supuesto podemos decir que en tal caso las maquinas $M_1, ..., M_{n+m}$ enumeran a S. La siguiente proposicion muestra que tambien las cosas se pueden hacer con una sola maquina de Turing.

Proposition 37 *Sea $S \subseteq \omega^n \times \Sigma^{*m}$ un conjunto no vacio. Entonces S es Σ-Turing enumerable si y solo si hay una maquina de Turing deterministica con unit $M = (Q, \Sigma, \Gamma, \delta, q_0, B, \shortmid, F)$, tal que:*

(1) Para cada $x \in \omega$, tenemos que M se detiene partiendo de $\lfloor q_0 B \shortmid^x \rfloor$ y llega a una descripcion instantanea de la forma $\lfloor qB \shortmid^{x_1} B...B \shortmid^{x_n} B\alpha_1 B...B\alpha_m \rfloor$, con $(\vec{x}, \vec{\alpha}) \in S$.

(2) Para cada $(\vec{x}, \vec{\alpha}) \in S$ hay un $x \in \omega$ tal que M se detiene partiendo de $\lfloor q_0 B \shortmid^x \rfloor$ y llega a una descripcion instantanea de la forma $\lfloor qB \shortmid^{x_1} B...B \shortmid^{x_n} B\alpha_1 B...B\alpha_m \rfloor$

Proof. Queda como ejercicio ver como construir la maquina M utilizando las maquinas $M_1, ..., M_{n+m}$ y reciprocamente ver como a partir de una maquina M con las propiedades (1) y (2) se pueden construir las maquinas $M_1, ..., M_{n+m}$. ■

3.1.5 Conjuntos Σ-Turing computables

La version Turingniana del concepto de conjunto Σ-efectivamente computable es facil de dar: un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ sera llamado Σ-*Turing computable* cuando la funcion $\chi_S^{\omega^n \times \Sigma^{*m}}$ sea Σ-Turing computable. O sea que $S \subseteq \omega^n \times \Sigma^{*m}$ es Σ-Turing computable sii hay una maquina de Turing deterministica con unit $M = (Q, \Sigma, \Gamma, \delta, q_0, B, \shortmid, F)$ la cual computa a $\chi_S^{\omega^n \times \Sigma^{*m}}$, es decir:

- Si $(\vec{x}, \vec{\alpha}) \in S$, entonces hay un $p \in Q$ tal que

$$\lfloor q_0 B \shortmid^{x_1} B ... B \shortmid^{x_n} B\alpha_1 B ... B\alpha_m \rfloor \overset{*}{\vdash} \lfloor pB \shortmid \rfloor$$

  y $\lfloor pB \shortmid \rfloor \nvdash d$, para cada $d \in Des$

- Si $(\vec{x}, \vec{\alpha}) \in (\omega^n \times \Sigma^{*m}) - S$, entonces hay un $p \in Q$ tal que

$$\lfloor q_0 B \shortmid^{x_1} B ... B \shortmid^{x_n} B\alpha_1 B ... B\alpha_m \rfloor \overset{*}{\vdash} \lfloor pB \rfloor$$

  y $\lfloor pB \rfloor \nvdash d$, para cada $d \in Des$

Si M es una maquina de Turing la cual computa a $\chi_S^{\omega^n \times \Sigma^{*m}}$, diremos que M *decide la pertenecia a* S, con respecto al conjunto $\omega^n \times \Sigma^{*m}$.

3.2 El paradigma de Godel: Funciones Σ-recursivas

En esta seccion desarrollaremos el modelo matematico del concepto de funcion Σ-efectivamente computable, dado por Godel. Dichas funciones seran llamadas Σ-recursivas. La idea es partir de un conjunto inicial de funciones muy simples y obviamente Σ-efectivamente computables y luego obtener nuevas funciones Σ-efectivamente computables usando constructores que preservan la computabilidad efectiva. Las funciones Σ-recursivas seran las que se obtienen iterando el uso de estos constructores, partiendo del conjunto inicial de funciones antes mencionado. Nos referiremos a este paradigma como el paradigma Godeliano o recursivo. A veces tambien lo llamaremos el paradigma funcional.

La familia de funciones simples y obviamente Σ-efectivamente computables de la que partiremos es la siguiente

$$\left\{Suc, Pred, C_0^{0,0}, C_\varepsilon^{0,0}\right\} \cup \{d_a : a \in \Sigma\} \cup \left\{p_j^{n,m} : 1 \leq j \leq n+m\right\}$$

Los constructores que usaremos son:

- Composicion
- Recursion primitiva
- Minimizacion de predicados totales

Estos constructores nos permiten dadas ciertas funciones construir o definir una nueva funcion y tienen la propiedad de preservar la computabilidad efectiva en el sentido que si las funciones iniciales son Σ-efectivamente computables, entonces la funcion obtenida tambien lo es. Un concepto fundamental es el de funcion Σ-recursiva primitiva. Estas funciones seran aquellas que se obtienen a partir de las del conjunto inicial usando solo los dos primeros constructores: composicion y recursion primitiva. Nuestro primer objetivo es definir el concepto de funcion Σ-recursiva primitiva para lo cual en las proximas dos secciones definiremos y estudiaremos los constructores de composicion y recursion primitiva. Luego definiremos el concepto de funcion Σ-recursiva primitiva y nos abocaremos a desarrollar este concepto fundamental. Recien despues estudiaremos el constructor de minimizacion y definiremos el concepto de funcion Σ-recursiva. La ultima parte de la seccion esta destinada a probar un teorema que nos dice que los conceptos de funcion Σ-recursiva y Σ-recursiva primitiva son independientes del alfabeto Σ.

3.2.1 Composicion

Dadas funciones Σ-mixtas $f, f_1, ..., f_r$, con $r \geq 1$, diremos que la funcion $f \circ [f_1, ..., f_r]$ es *obtenida por composicion a partir de las funciones* $f, f_1, ..., f_r$. Para probar que la composicion preserva la computabilidad efectiva necesitaremos el siguiente lema.

Lemma 38 *Supongamos que $f, f_1, ..., f_r$ son funciones Σ-mixtas, con $r \geq 1$. Supongamos ademas que $f \circ [f_1, ..., f_r] \neq \emptyset$. Entonces hay $n, m, k, l \in \omega$ y $s \in \{\#, *\}$ tales que*

- *$r = n + m$*
- *f es de tipo (n, m, s)*
- *f_i es de tipo $(k, l, \#)$, para cada $i = 1, ..., n$*
- *f_i es de tipo $(k, l, *)$, para cada $i = n + 1, ..., n + m$*

Mas aun, en tal caso la funcion $f \circ [f_1, ..., f_{n+m}]$ es de tipo (k, l, s) y:

$$\begin{aligned} D_{f\circ[f_1,...,f_{n+m}]} &= \left\{ (\vec{x}, \vec{\alpha}) \in \bigcap_{i=1}^{n+m} D_{f_i} : (f_1(\vec{x}, \vec{\alpha}), ..., f_{n+m}(\vec{x}, \vec{\alpha})) \in D_f \right\} \\ f \circ [f_1, ..., f_{n+m}](\vec{x}, \vec{\alpha}) &= f(f_1(\vec{x}, \vec{\alpha}), ..., f_{n+m}(\vec{x}, \vec{\alpha})). \end{aligned}$$

Proof. Notese que $f \neq \emptyset$ y $[f_1, ..., f_r] \neq \emptyset$ (por que?). Ya que $f \neq \emptyset$ tenemos que hay unicos $n, m \in \omega$ y $s \in \{\#, *\}$ tales que f es de tipo (n, m, s). Ya que $f \circ [f_1, ..., f_r] \neq \emptyset$ y $I_{[f_1,...,f_r]} \subseteq I_{f_1} \times ... \times I_{f_r}$, tenemos que

- $r = n + m$

- $I_{f_i} \subseteq \omega$, para cada $i = 1, ..., n$

- $I_{f_i} \subseteq \Sigma^*$, para cada $i = n+1, ..., n+m$

Ya que $[f_1, ..., f_r] \neq \emptyset$ tenemos que $D_{[f_1,...,f_r]} = \bigcap_{i=1}^r D_{f_i} \neq \emptyset$, por lo cual los conjuntos $D_{f_1}, ..., D_{f_{n+m}}$ deberan ser todos de un mismo tipo, digamos de tipo (k, l). Es decir que f_i es de tipo $(k, l, \#)$, para cada $i = 1, ..., n$ y f_i es de tipo $(k, l, *)$, para cada $i = n+1, ..., n+m$.

Las ultimas observaciones del lema son directas de las definiciones de $[f_1, ..., f_{n+m}]$ y de composicion de funciones ■

Ahora si podemos probar facilmente que el contructor composicion preserva la computabilidad efectiva

Lemma 39 *Si $f, f_1, ..., f_r$, con $r \geq 1$, son Σ-efectivamente computables, entonces $f \circ [f_1, ..., f_r]$ lo es.*

Proof. Si $f \circ [f_1, ..., f_r] = \emptyset$, entonces claramente es Σ-efectivamente computable. Supongamos entonces que $f \circ [f_1, ..., f_r] \neq \emptyset$. Por el lema anterior hay $n, m, k, l \in \omega$ y $s \in \{\#, *\}$ tales que

- $r = n + m$

- f es de tipo (n, m, s)

- f_i es de tipo $(k, l, \#)$, para cada $i = 1, ..., n$

- f_i es de tipo $(k, l, *)$, para cada $i = n+1, ..., n+m$

Sean $\mathbb{P}, \mathbb{P}_1, ..., \mathbb{P}_{n+m}$ procedimientos efectivos los cuales computen las funciones $f, f_1, ..., f_{n+m}$, respectivamente. Usando estos procedimientos es facil definir un procedimiento efectivo el cual compute a $f \circ [f_1, ..., f_{n+m}]$. ■

3.2.2 Recursion primitiva

La recursion primitiva es un tipo muy particular de recursion. Consideremos por ejemplo las siguientes ecuaciones:

(1) $R(0) = 1$

(2) $R(t + 1) = 1 + R(t) + R(t)^2$

Notese que hay una unica funcion $R : \omega \to \omega$ la cual cumple (1) y (2). Esto es ya que el valor de R en t esta determinado por sucesivas aplicaciones de las ecuaciones (1) y (2). Por ejemplo la ecuacion (1) nos dice que $R(0) = 1$ pero entonces la ecuacion (2) nos dice que $R(1) = 1 + 1 + 1^2 = 3$ por lo cual nuevamente la ecuacion (2) nos dice que $R(2) = 1 + 3 + 3^2 = 13$ y asi podemos notar facilmente que R esta determinada por dichas ecuaciones.

Se suele decir que las ecuaciones (1) y (2) definen recursivamente a la funcion R pero hay que tener cuidado porque esto es una manera de hablar ya que la funcion R podria en nuestro discurso ya haber sido definida de otra manera. Mas propio es pensar que dichas ecuaciones determinan a R en el sentido que R es la unica que las cumple. Por ejemplo las ecuaciones:

(a) $R(0) = 50$

(b) $R(t + 1) = R(t)$

definen recursivamente a la funcion $C_{50}^{1,0}$ pero esta claro que la definicion de $C_{50}^{1,0}$ en esta materia no fue dada de esta forma.

Hay casos de recursiones en las cuales el valor de $R(t + 1)$ no solo depende de $R(t)$ sino que tambien depende de t. Por ejemplo

(i) $R(0) = 1$

(ii) $R(t + 1) = t.R(t) + 1$

De todas maneras deberia quedar claro que las ecuaciones (i) y (ii) determinan una unica funcion $R : \omega \to \omega$ que las satisface.

Tambien podemos generalizar pensando que la funcion R depende no solo de un parametro t sino que su dominio es ω^4, es decir depende de t y x_1, x_2, x_3. Por ejemplo

(p) $R(0, x_1, x_2, x_3) = x_1 + 2x_3$

(q) $R(t + 1, x_1, x_2, x_3) = t + x_1 + x_2 + x_3 + R(t, x_1, x_2, x_3)$

Dejamos al lector convencerse de que (p) y (q) son cumplidas por una unica funcion $R : \omega^4 \to \omega$. Tambien podriamos tener variables alfabeticas. Por ejemplo consideremos

(r) $R(0, x_1, x_2, \alpha_1, \alpha_2) = x_1 + |\alpha_1|^{x_2}$

(s) $R(t + 1, x_1, x_2, \alpha_1, \alpha_2) = t + x_1 + x_2 + |\alpha_1| + |\alpha_2| + R(t, x_1, x_2, \alpha_1, \alpha_2)$

Es claro aqui que las ecuaciones (r) y (s) determinan una unica funcion $R : \omega^3 \times \Sigma^{*2} \to \omega$ que las cumple. Esto se puede explicar de la siguiente manera:

- La ecuacion (r) determina los valores de R sobre el conjunto $\{0\} \times \omega \times \omega \times \Sigma^* \times \Sigma^*$. Pero una ves determinados estos valores, la ecuacion (s) tomada con $t = 0$, determina los valores de R sobre el conjunto $\{1\} \times \omega \times \omega \times \Sigma^* \times \Sigma^*$. Pero una ves determinados estos valores, la ecuacion (s) tomada con $t = 1$, determina los valores de R sobre el conjunto $\{2\} \times \omega \times \omega \times \Sigma^* \times \Sigma^*$, etc

El caso anterior podria generalizarse de la siguiente manera: Si tenemos dadas dos funciones

$$\begin{array}{rcl} f & : & \omega^n \times \Sigma^{*m} \to \omega \\ g & : & \omega^{n+2} \times \Sigma^{*m} \to \omega \end{array}$$

entonces las ecuaciones:

(a) $R(0,\vec{x},\vec{\alpha}) = f(\vec{x},\vec{\alpha})$

(b) $R(t+1,\vec{x},\vec{\alpha}) = g(R(t,\vec{x},\vec{\alpha}),t,\vec{x},\vec{\alpha})$

determinan una unica funcion $R : \omega^{n+1} \times \Sigma^{*m} \to \omega$ que las cumple. Notese que para el caso

$$\begin{array}{rcl} n & = & m = 2 \\ f & = & \lambda x_1 x_2 \alpha_1 \alpha_2 [x_1 + |\alpha_1|^{x_2}] \\ g & = & \lambda x t x_1 x_2 \alpha_1 \alpha_2 [t + x_1 + x_2 + |\alpha_1| + |\alpha_2| + x] \end{array}$$

las ecuaciones (a) y (b) se transforman en las ecuaciones (r) y (s).

El primer caso de recursion primitiva que definiremos a continuacion engloba los ejemplos vistos recien dentro de un marco general.

Recursion primitiva sobre variable numerica con valores numericos Supongamos tenemos dadas funciones

$$\begin{array}{rcl} f & : & S_1 \times ... \times S_n \times L_1 \times ... \times L_m \to \omega \\ g & : & \omega \times \omega \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m \to \omega \end{array}$$

con $S_1, ..., S_n \subseteq \omega$ y $L_1, ..., L_m \subseteq \Sigma^*$ conjuntos no vacios. Usando el razonamiento inductivo usado en los ejemplos anteriores, se puede probar que hay una unica funcion

$$R : \omega \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m \to \omega$$

la cual cumple las ecuaciones

- $R(0,\vec{x},\vec{\alpha}) = f(\vec{x},\vec{\alpha})$
- $R(t+1,\vec{x},\vec{\alpha}) = g(R(t,\vec{x},\vec{\alpha}),t,\vec{x},\vec{\alpha})$

LLamaremos $R(f,g)$ a esta unica funcion que cumple las ecuaciones anteriores. Resumiendo, diremos que las ecuaciones

(1) $R(f,g)(0,\vec{x},\vec{\alpha}) = f(\vec{x},\vec{\alpha})$

(2) $R(f,g)(t+1,\vec{x},\vec{\alpha}) = g(R(f,g)(t,\vec{x},\vec{\alpha}),t,\vec{x},\vec{\alpha})$

definen recursivamente a la funcion $R(f,g)$. Tambien diremos que $R(f,g)$ es obtenida por *recursion primitiva* a partir de f y g.

NOTA IMPOTANTE: No confundirse y pensar que $R(f,g)$ es el resultado de aplicar una funcion R al par (f,g), de hecho hasta el momento no hemos definido ninguna funcion R cuyo dominio sea cierto conjunto de pares ordenados de funciones!

Notese que cuando $m = n = 0$, se tiene que $D_f = \{\Diamond\}$ y (1) y (2) se transforman en

(1) $R(f,g)(0) = f(\Diamond)$

(2) $R(f,g)(t+1) = g(R(f,g)(t),t)$

Veamos algunos ejemplos

E$_1$ Tomemos $f = p_1^{1,0}$ y $g = Suc \circ p_1^{3,0}$. De la definicion de $R(f,g)$, obtenemos que su dominio es ω^2 y

$$\begin{aligned} R(f,g)(0,x_1) &= p_1^{1,0}(x_1) = x_1 \\ R(f,g)(t+1,x_1) &= \left(Suc \circ p_1^{3,0}\right)(R(f,g)(t,x_1),t,x_1) = R(f,g)(t,x_1)+1 \end{aligned}$$

Es facil notar que la unica funcion que cumple estas dos ecuaciones es $\lambda t x_1 [t+x_1]$, lo cual implica que $\lambda t x_1 [t+x_1] = R\left(p_1^{1,0}, Suc \circ p_1^{3,0}\right)$

E$_2$ Sean $f = C_0^{0,0}$ y $g = p_1^{2,0}$. De la definicion de $R(f,g)$, obtenemos que su dominio es ω y

$$\begin{aligned} R(f,g)(0) &= C_0^{0,0}(\Diamond) = 0 \\ R(f,g)(t+1) &= p_1^{2,0}(R(f,g)(t),t) = R(f,g)(t) \end{aligned}$$

Es facil notar que la unica funcion que cumple estas dos ecuaciones es $C_0^{1,0}$ lo cual implica que $C_0^{1,0} = R\left(C_0^{0,0}, p_1^{2,0}\right)$

Como era de esperar, este caso del constructor de recursion primitiva preserva la computabilidad efectiva

Lemma 40 *Si f y g son Σ-efectivamente computables, entonces $R(f,g)$ lo es.*

Proof. Es dejada al lector ■

Nota importante: En los ejemplos anteriores y en todos los casos que manejaremos en esta primera etapa, en las aplicaciones del constructor de recursion primitiva (en sus cuatro formas) las funciones iniciales seran Σ-totales (es decir $S_1 = ... = S_n = \omega$ y $L_1 = ... = L_m = \Sigma^*$). Mas adelante veremos aplicaciones con funciones no Σ-totales.

Recursion primitiva sobre variable numerica con valores alfabeticos Ahora haremos el caso en el que la funcion definida recursivamente tiene imagen contenida en Σ^*. Es claro que entonces f y g tambien deberan tener imagen contenida en Σ^*. El unico detalle a tener en cuenta en la definicion de este caso es que si solo hicieramos estos cambios y pusieramos las mismas ecuaciones la funcion g no resultaria Σ-mixta en general. Para que la g de la recursion siga siendo Σ-mixta deberemos modificar levemente su dominio en relacion al caso ya hecho

Supongamos Σ es un alfabeto finito. Sean

$$\begin{array}{lll} f & : & S_1 \times ... \times S_n \times L_1 \times ... \times L_m \rightarrow \Sigma^* \\ g & : & \omega \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m \times \Sigma^* \rightarrow \Sigma^* \end{array}$$

con $S_1, ..., S_n \subseteq \omega$ y $L_1, ..., L_m \subseteq \Sigma^*$ conjuntos no vacios. Definamos

$$R(f,g) : \omega \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m \rightarrow \Sigma^*$$

de la siguiente manera

(1) $R(f,g)(0,\vec{x},\vec{\alpha}) = f(\vec{x},\vec{\alpha})$

(2) $R(f,g)(t+1,\vec{x},\vec{\alpha}) = g(t,\vec{x},\vec{\alpha},R(f,g)(t,\vec{x},\vec{\alpha}))$

Diremos que $R(f,g)$ es obtenida por *recursion primitiva* a partir de f y g. Notese que cuando $m = n = 0$, se tiene que $D_f = \{\Diamond\}$ y (1) y (2) se transforman en

(1) $R(f,g)(0) = f(\Diamond)$

(2) $R(f,g)(t+1) = g(t,R(f,g)(t))$

Veamos algunos ejemplos

E$_1$ Tomemos $f = C_\varepsilon^{0,1}$ y $g = \lambda\alpha\beta\left[\alpha\beta\right] \circ \left[p_3^{1,2}, p_2^{1,2}\right]$. De la definicion de $R(f,g)$, obtenemos que

$$\begin{array}{rcl} R(f,g)(0,\alpha_1) & = & C_\varepsilon^{0,1}(\alpha_1) = \varepsilon \\ R(f,g)(t+1,\alpha_1) & = & \lambda\alpha\beta\left[\alpha\beta\right] \circ \left[p_3^{1,2}, p_2^{1,2}\right](t,\alpha_1,R(f,g)(t,\alpha_1)) = R(f,g)(t,\alpha_1)\alpha_1 \end{array}$$

Es facil notar que la unica funcion que cumple estas dos ecuaciones es $\lambda t\alpha_1\left[{\alpha_1}^t\right]$, lo cual implica que $\lambda t\alpha_1\left[{\alpha_1}^t\right] = R\left(C_\varepsilon^{0,1}, \lambda\alpha\beta\left[\alpha\beta\right] \circ \left[p_3^{1,2}, p_2^{1,2}\right]\right)$

E$_2$ Sean $f = C_\varepsilon^{0,0}$ y $g = p_2^{2,0}$. De la definicion de $R(f,g)$, obtenemos que

$$\begin{array}{rcl} R(f,g)(0) & = & C_\varepsilon^{0,0}(\Diamond) = \varepsilon \\ R(f,g)(t+1) & = & p_2^{2,0}(t,R(f,g)(t)) = R(f,g)(t) \end{array}$$

Es facil notar que la unica funcion que cumple estas dos ecuaciones es $C_\varepsilon^{1,0}$ lo cual implica que $C_\varepsilon^{1,0} = R\left(C_\varepsilon^{0,0}, p_2^{2,0}\right)$

La prueba del siguiente lema es completamente analoga a la del lema anterior que fue dejada como ejercicio.

Lemma 41 *Si f y g son Σ-efectivamente computables, entonces $R(f,g)$ lo es.*

Recursion primitiva sobre variable alfabetica con valores numericos Ya vimos dos casos de recursion donde el parametro que comanda la recursion es numerico. Daremos a continuacion un ejemplo de recursion en el cual el parametro principal es alfabetico. Sea $\Sigma = \{\%, @, ?\}$ y consideremos las siguientes ecuaciones:

(1) $R(\varepsilon) = 15$

(2) $R(\alpha\%) = R(\alpha) + 1$

(3) $R(\alpha@) = R(\alpha).5$

(4) $R(\alpha?) = R(\alpha)^{20}$

Notese que las ecuaciones anteriores determinan una funcion $R : \Sigma^* \to \omega$. Esto es ya que R en ε debe valer 15 y sabiendo esto las ecuaciones (2), (3) y (4) (con $\alpha = \varepsilon$) nos dicen que

$$\begin{array}{rcl} R(\%) & = & 16 \\ R(@) & = & 75 \\ R(?) & = & 15^{20} \end{array}$$

por lo cual podemos aplicarlas nuevamente a dichas ecuaciones (con $\alpha \in \{\%, @, ?\}$) para calcular R en todas las palabras de longitud 2; y asi sucesivamente.

Daremos otro ejemplo un poco mas complicado para seguir aproximandonos al caso general. Nuevamente supongamos que $\Sigma = \{\%, @, ?\}$ y supongamos tenemos una funcion

$$f : \omega \times \Sigma^* \to \omega$$

y tres funciones

$$\begin{array}{rcl} \mathcal{G}_{\%} & : & \omega \times \omega \times \Sigma^* \times \Sigma^* \to \omega \\ \mathcal{G}_{@} & : & \omega \times \omega \times \Sigma^* \times \Sigma^* \to \omega \\ \mathcal{G}_{?} & : & \omega \times \omega \times \Sigma^* \times \Sigma^* \to \omega \end{array}$$

Entonces hay una unica funcion $R :: \omega \times \Sigma^* \times \Sigma^* \to \omega$ la cual cumple las siguientes ecuaciones

(1) $R(x_1, \alpha_1, \varepsilon) = f(x_1, \alpha_1)$

(2) $R(x_1, \alpha_1, \alpha\%) = \mathcal{G}_{\%}(R(x_1, \alpha_1, \alpha), x_1, \alpha_1, \alpha)$

(3) $R(x_1, \alpha_1, \alpha@) = \mathcal{G}_@(R(x_1, \alpha_1, \alpha), x_1, \alpha_1, \alpha)$

(4) $R(x_1, \alpha_1, \alpha?) = \mathcal{G}_?(R(x_1, \alpha_1, \alpha), x_1, \alpha_1, \alpha)$

(Justifique que las ecuaciones anteriores determinan a la funcion R.)

El ejemplo anterior nos muestra que para hacer recursion sobre parametro alfabetico nos hace falta "una funcion g por cada simbolo de Σ". Esto motiva la siguiente definicion. Dado un alfabeto Σ, una *familia Σ-indexada de funciones* sera una funcion $\mathcal{G}$ tal que $D_\mathcal{G} = \Sigma$ y para cada $a \in D_\mathcal{G}$ se tiene que $\mathcal{G}(a)$ es una funcion. Algunos ejemplos:

E$_1$ Sea $\mathcal{G}$ dada por

$$\begin{array}{rcl} \mathcal{G} : \{\square, \%, \blacktriangle\} & \rightarrow & \{Suc, Pred\} \\ \square & \rightarrow & Suc \\ \% & \rightarrow & Suc \\ \blacktriangle & \rightarrow & Pred \end{array}$$

Claramente $\mathcal{G}$ es una familia $\{\square, \%, \blacktriangle\}$-indexada de funciones. Notar que

$$\mathcal{G} = \{(\square, Suc), (\%, Suc), (\blacktriangle, Pred)\}$$

Se tiene tambien por ejemplo que $\mathcal{G}(\%) = Suc$ por lo cual tambien es cierto que $\mathcal{G}(\%)(22) = 23$, etc.

E$_2$ Si Σ es un alfabeto no vacio, la funcion

$$\begin{array}{rcl} \mathcal{G} : \Sigma & \rightarrow & \{f : f \text{ es una funcion de } \Sigma^* \text{ en } \Sigma^*\} \\ a & \rightarrow & d_a \end{array}$$

es una familia Σ-indexada de funciones. Notar que

$$\mathcal{G} = \{(a, d_a) : a \in \Sigma\}$$

E$_3$ $\emptyset$ es una flia $\emptyset$-indexada de funciones

Si $\mathcal{G}$ es una familia Σ-indexada de funciones, entonces para $a \in \Sigma$, escribiremos $\mathcal{G}_a$ en lugar de $\mathcal{G}(a)$. Ahora sí, nuestro caso de recursion primitiva. Sea

$$f : S_1 \times ... \times S_n \times L_1 \times ... \times L_m \rightarrow \omega$$

con $S_1, ..., S_n \subseteq \omega$ y $L_1, ..., L_m \subseteq \Sigma^*$ conjuntos no vacios y sea $\mathcal{G}$ una familia Σ-indexada de funciones tal que

$$\mathcal{G}_a : \omega \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m \times \Sigma^* \rightarrow \omega$$

para cada $a \in \Sigma$. Definamos

$$R(f, \mathcal{G}) : S_1 \times ... \times S_n \times L_1 \times ... \times L_m \times \Sigma^* \rightarrow \omega$$

de la siguiente manera

(1) $R(f,\mathcal{G})(\vec{x},\vec{\alpha},\varepsilon) = f(\vec{x},\vec{\alpha})$

(2) $R(f,\mathcal{G})(\vec{x},\vec{\alpha},\alpha a) = \mathcal{G}_a(R(f,\mathcal{G})(\vec{x},\vec{\alpha},\alpha),\vec{x},\vec{\alpha},\alpha)$

Diremos que $R(f,\mathcal{G})$ es obtenida por *recursion primitiva* a partir de f y $\mathcal{G}$. Notese que cuando $m = n = 0$, se tiene que $D_f = \{\Diamond\}$ y (1) y (2) se transforman en

(1) $R(f,\mathcal{G})(\varepsilon) = f(\Diamond)$

(2) $R(f,\mathcal{G})(\alpha a) = \mathcal{G}_a(R(f,\mathcal{G})(\alpha),\alpha)$

Lemma 42 *Si f y cada $\mathcal{G}_a$ son Σ-efectivamente computables, entonces $R(f,\mathcal{G})$ lo es.*

Proof. Es dejada al lector ■

Recursion primitiva sobre variable alfabetica con valores alfabeticos Supongamos Σ es un alfabeto finito. Sea

$$f : S_1 \times ... \times S_n \times L_1 \times ... \times L_m \rightarrow \Sigma^*$$

con $S_1,...,S_n \subseteq \omega$ y $L_1,...,L_m \subseteq \Sigma^*$ conjuntos no vacios y sea $\mathcal{G}$ una familia Σ-indexada de funciones tal que

$$\mathcal{G}_a : S_1 \times ... \times S_n \times L_1 \times ... \times L_m \times \Sigma^* \times \Sigma^* \rightarrow \Sigma^*$$

para cada $a \in \Sigma$. Definamos

$$R(f,\mathcal{G}) : S_1 \times ... \times S_n \times L_1 \times ... \times L_m \times \Sigma^* \rightarrow \Sigma^*$$

de la siguiente manera

(1) $R(f,\mathcal{G})(\vec{x},\vec{\alpha},\varepsilon) = f(\vec{x},\vec{\alpha})$

(2) $R(f,\mathcal{G})(\vec{x},\vec{\alpha},\alpha a) = \mathcal{G}_a(\vec{x},\vec{\alpha},\alpha,R(f,\mathcal{G})(\vec{x},\vec{\alpha},\alpha))$.

Diremos que $R(f,\mathcal{G})$ es obtenida por *recursion primitiva* a partir de f y $\mathcal{G}$. Notese que cuando $m = n = 0$, se tiene que $D_f = \{\Diamond\}$ y (1) y (2) se transforman en

(1) $R(f,\mathcal{G})(\varepsilon) = f(\Diamond)$

(2) $R(f,\mathcal{G})(\alpha a) = \mathcal{G}_a(\alpha,R(f,\mathcal{G})(\alpha))$

La prueba del siguiente lema es completamente analoga a la del lema anterior que fue dejada como ejercicio.

Lemma 43 *Si f y cada $\mathcal{G}_a$ son Σ-efectivamente computables, entonces $R(f,\mathcal{G})$ lo es.*

3.2.3 Funciones Σ-recursivas primitivas

Intuitivamente hablando una funcion es Σ-recursiva primitiva si se puede obtener de las iniciales usando los constructores de composicion y recursion primitiva. Daremos ahora una definicion matematica de este concepto. Definamos los conjuntos $\mathrm{PR}_0^\Sigma \subseteq \mathrm{PR}_1^\Sigma \subseteq \mathrm{PR}_2^\Sigma \subseteq ... \subseteq \mathrm{PR}^\Sigma$ de la siguiente manera

$$\begin{array}{lll}
\mathrm{PR}_0^\Sigma & = & \left\{Suc, Pred, C_0^{0,0}, C_\varepsilon^{0,0}\right\} \cup \{d_a : a \in \Sigma\} \cup \left\{p_j^{n,m} : 1 \leq j \leq n+m\right\} \\
\mathrm{PR}_{k+1}^\Sigma & = & \mathrm{PR}_k^\Sigma \cup \left\{f \circ [f_1, ..., f_r] : f, f_1, ..., f_r \in \mathrm{PR}_k^\Sigma,\, r \geq 1\right\} \cup \\
 & & \qquad \left\{R(f, \mathcal{G}) : f \text{ y cada } \mathcal{G}_a \text{ pertenecen a } \mathrm{PR}_k^\Sigma\right\} \cup \\
 & & \qquad \qquad \left\{R(f,g) : f, g \in \mathrm{PR}_k^\Sigma\right\} \\
\mathrm{PR}^\Sigma & = & \bigcup_{k \geq 0} \mathrm{PR}_k^\Sigma
\end{array}$$

Una funcion es llamada Σ-*recursiva primitiva* (Σ-*p.r.*) si pertenece a PR^Σ.

Proposition 44 *Si $f \in \mathrm{PR}^\Sigma$, entonces f es Σ-efectivamente computable.*

Proof. Dejamos al lector la prueba por induccion en k de que si $f \in \mathrm{PR}_k^\Sigma$, entonces f es Σ-efectivamente computable, la cual sale en forma directa usando los lemas anteriores que garantizan que los constructores de composicion y recursion primitiva preservan la computabilidad efectiva ■

Algunas funciones Σ-recursivas primitivas En los siguientes cuatro lemas se prueba bien formalmente que varias funciones bien conocidas son Σ-primitivas recursivas.

Lemma 45 *Sea Σ un alfabeto finito.*

(1) $\emptyset \in \mathrm{PR}^\Sigma$.

(2) $\lambda xy\,[x+y] \in \mathrm{PR}^\Sigma$.

(3) $\lambda xy\,[x.y] \in \mathrm{PR}^\Sigma$.

(4) $\lambda x\,[x!] \in \mathrm{PR}^\Sigma$.

Proof. (1) Notese que $\emptyset = Pred \circ C_0^{0,0} \in \mathrm{PR}_1^\Sigma$

(2) Notar que

$$\begin{array}{rcl}
\lambda xy\,[x+y]\,(0, x_1) & = & x_1 = p_1^{1,0}(x_1) \\
\lambda xy\,[x+y]\,(t+1, x_1) & = & \lambda xy\,[x+y]\,(t, x_1) + 1 \\
 & = & \left(Suc \circ p_1^{3,0}\right)(\lambda xy\,[x+y]\,(t, x_1), t, x_1)
\end{array}$$

lo cual implica que $\lambda xy\,[x+y] = R\left(p_1^{1,0}, Suc \circ p_1^{3,0}\right) \in \mathrm{PR}_2^\Sigma$.

(3) Primero note que

$$\begin{array}{rcl} C_0^{1,0}(0) & = & C_0^{0,0}(\Diamond) \\ C_0^{1,0}(t+1) & = & C_0^{1,0}(t) \end{array}$$

lo cual implica que $C_0^{1,0} = R\left(C_0^{0,0}, p_1^{2,0}\right) \in \mathrm{PR}_1^\Sigma$. Tambien note que

$$\lambda tx\,[t.x] = R\left(C_0^{1,0}, \lambda xy\,[x+y] \circ \left[p_1^{3,0}, p_3^{3,0}\right]\right),$$

lo cual por (2) implica que $\lambda tx\,[t.x] \in \mathrm{PR}_4^\Sigma$.

(4) Note que

$$\begin{array}{rcl} \lambda x\,[x!]\,(0) & = & 1 = C_1^{0,0}(\Diamond) \\ \lambda x\,[x!]\,(t+1) & = & \lambda x\,[x!]\,(t).(t+1), \end{array}$$

lo cual implica que

$$\lambda x\,[x!] = R\left(C_1^{0,0}, \lambda xy\,[x.y] \circ \left[p_1^{2,0}, Suc \circ p_2^{2,0}\right]\right).$$

Ya que $C_1^{0,0} = Suc \circ C_0^{0,0}$, tenemos que $C_1^{0,0} \in \mathrm{PR}_1^\Sigma$. Por (3), tenemos que

$$\lambda xy\,[x.y] \circ \left[p_1^{2,0}, Suc \circ p_2^{2,0}\right] \in \mathrm{PR}_5^\Sigma,$$

obteniendo que $\lambda x\,[x!] \in \mathrm{PR}_6^\Sigma$. ■

Ahora consideraremos dos funciones las cuales son obtenidas naturalmente por recursion primitiva sobre variable alfabetica.

Lemma 46 *Supongamos Σ es un alfabeto finito.*

(a) $\lambda\alpha\beta\,[\alpha\beta] \in \mathrm{PR}^\Sigma$

(b) $\lambda\alpha\,[|\alpha|] \in \mathrm{PR}^\Sigma$

Proof. (a) Ya que

$$\begin{array}{rcl} \lambda\alpha\beta\,[\alpha\beta]\,(\alpha_1,\varepsilon) & = & \alpha_1 = p_1^{0,1}(\alpha_1) \\ \lambda\alpha\beta\,[\alpha\beta]\,(\alpha_1,\alpha a) & = & d_a(\lambda\alpha\beta\,[\alpha\beta]\,(\alpha_1,\alpha)), a \in \Sigma \end{array}$$

tenemos que $\lambda\alpha\beta\,[\alpha\beta] = R\left(p_1^{0,1}, \mathcal{G}\right)$, donde $\mathcal{G}_a = d_a \circ p_3^{0,3}$, para cada $a \in \Sigma$.

(b) Ya que

$$\begin{array}{rcl} \lambda\alpha\,[|\alpha|]\,(\varepsilon) & = & 0 = C_0^{0,0}(\Diamond) \\ \lambda\alpha\,[|\alpha|]\,(\alpha a) & = & \lambda\alpha\,[|\alpha|]\,(\alpha) + 1 \end{array}$$

tenemos que $\lambda\alpha\,[|\alpha|] = R\left(C_0^{0,0}, \mathcal{G}\right)$, donde $\mathcal{G}_a = Suc \circ p_1^{1,1}$, para cada $a \in \Sigma$..
■

Lemma 47 *Sea Σ un alfabeto finito. Entonces $C_k^{n,m}, C_\alpha^{n,m} \in \mathrm{PR}^\Sigma$, para cada $n, m, k \geq 0$ y $\alpha \in \Sigma^*$.*

Proof. (a) Note que $C_{k+1}^{0,0} = Suc \circ C_k^{0,0}$, lo cual implica $C_k^{0,0} \in \mathrm{PR}_k^\Sigma$, para $k \geq 0$. Tambien note que $C_{\alpha a}^{0,0} = d_a \circ C_\alpha^{0,0}$, lo cual dice que $C_\alpha^{0,0} \in \mathrm{PR}^\Sigma$, para $\alpha \in \Sigma^*$. Para ver que $C_k^{0,1} \in \mathrm{PR}^\Sigma$ notar que

$$\begin{array}{rcl} C_k^{0,1}(\varepsilon) & = & k = C_k^{0,0}(\Diamond) \\ C_k^{0,1}(\alpha a) & = & C_k^{0,1}(\alpha) = p_1^{1,1}\left(C_k^{0,1}(\alpha), \alpha\right) \end{array}$$

lo cual implica que $C_k^{0,1} = R\left(C_k^{0,0}, \mathcal{G}\right)$, con $\mathcal{G}_a = p_1^{1,1}$, $a \in \Sigma$. En forma similar podemos ver que $C_k^{1,0}, C_\alpha^{1,0}, C_\alpha^{0,1} \in \mathrm{PR}^\Sigma$. Supongamos ahora que $m > 0$. Entonces

$$\begin{array}{rcl} C_k^{n,m} & = & C_k^{0,1} \circ p_{n+1}^{n,m} \\ C_\alpha^{n,m} & = & C_\alpha^{0,1} \circ p_{n+1}^{n,m} \end{array}$$

de lo cual obtenemos que $C_k^{n,m}, C_\alpha^{n,m} \in \mathrm{PR}^\Sigma$. El caso $n > 0$ es similar. ■

Lemma 48 *Sea Σ un alfabeto finito.*

(a) $\lambda xy\,[x^y] \in \mathrm{PR}^\Sigma$.

(b) $\lambda t\alpha\,[\alpha^t] \in \mathrm{PR}^\Sigma$.

Proof. (a) Note que

$$\lambda tx\left[x^t\right] = R\left(C_1^{1,0}, \lambda xy\,[x.y] \circ \left[p_1^{3,0}, p_3^{3,0}\right]\right) \in \mathrm{PR}^\Sigma.$$

O sea que $\lambda xy\,[x^y] = \lambda tx\,[x^t] \circ \left[p_2^{2,0}, p_1^{2,0}\right] \in \mathrm{PR}^\Sigma$.

(b) Note que

$$\lambda t\alpha\left[\alpha^t\right] = R\left(C_\varepsilon^{0,1}, \lambda\alpha\beta\,[\alpha\beta] \circ \left[p_3^{1,2}, p_2^{1,2}\right]\right) \in \mathrm{PR}^\Sigma.$$

■

Ahora probaremos que si Σ es no vacio, entonces las biyeciones naturales entre Σ^* y ω, dadas en el Lema **??**, son Σ-p.r..

Lemma 49 *Si $\leq$ es un orden total sobre un alfabeto no vacio Σ, entonces $s^{\leq}$, $\#^{\leq}$ y $*^{\leq}$ pertenecen a PR^Σ*

Proof. Supongamos $\Sigma = \{a_1, ..., a_k\}$ y $\leq$ es dado por $a_1 < ... < a_k$. Ya que

$$\begin{array}{rcl} s^{\leq}(\varepsilon) & = & a_1 \\ s^{\leq}(\alpha a_i) & = & \alpha a_{i+1}, \text{ para } i < k \\ s^{\leq}(\alpha a_k) & = & s^{\leq}(\alpha) a_1 \end{array}$$

tenemos que $s^{\leq} = R\left(C_{a_1}^{0,0}, \mathcal{G}\right)$, donde $\mathcal{G}_{a_i} = d_{a_{i+1}} \circ p_1^{0,2}$, para $i = 1, ..., k-1$ y $\mathcal{G}_{a_k} = d_{a_1} \circ p_2^{0,2}$. O sea que $s^{\leq} \in \mathrm{PR}^{\Sigma}$. Ya que

$$\begin{array}{rcl} *^{\leq}(0) & = & \varepsilon \\ *^{\leq}(t+1) & = & s^{\leq}(*^{\leq}(t)) \end{array}$$

podemos ver que $*^{\leq} \in \mathrm{PR}^{\Sigma}$. Ya que

$$\begin{array}{rcl} \#^{\leq}(\varepsilon) & = & 0 \\ \#^{\leq}(\alpha a_i) & = & \#^{\leq}(\alpha).k + i, \text{ para } i = 1, ..., k, \end{array}$$

tenemos que $\#^{\leq} = R\left(C_0^{0,0}, \mathcal{G}\right)$, donde

$$\mathcal{G}_{a_i} = \lambda xy\left[x+y\right] \circ \left[\lambda xy\left[x.y\right] \circ \left[p_1^{1,1}, C_k^{1,1}\right], C_i^{1,1}\right], \text{ para } i = 1, ..., k.$$

O sea que $\#^{\leq} \in \mathrm{PR}^{\Sigma}$. ■

Dados $x, y \in \omega$, definamos

$$x \dot{-} y = \max(x-y, 0).$$

Lemma 50 *(a)* $\lambda xy\left[x \dot{-} y\right] \in \mathrm{PR}^{\Sigma}$.

(b) $\lambda xy\left[\max(x,y)\right] \in \mathrm{PR}^{\Sigma}$.

(c) $\lambda xy\left[x = y\right] \in \mathrm{PR}^{\Sigma}$.

(d) $\lambda xy\left[x \leq y\right] \in \mathrm{PR}^{\Sigma}$.

(e) $\lambda \alpha\beta\left[\alpha = \beta\right] \in \mathrm{PR}^{\Sigma}$

Proof. (a) Primero notar que $\lambda x\left[x \dot{-} 1\right] = R\left(C_0^{0,0}, p_2^{2,0}\right) \in \mathrm{PR}^{\Sigma}$. Tambien note que

$$\lambda tx\left[x \dot{-} t\right] = R\left(p_1^{1,0}, \lambda x\left[x \dot{-} 1\right] \circ p_1^{3,0}\right) \in \mathrm{PR}^{\Sigma}.$$

O sea que $\lambda xy\left[x \dot{-} y\right] = \lambda tx\left[x \dot{-} t\right] \circ \left[p_2^{2,0}, p_1^{2,0}\right] \in \mathrm{PR}^{\Sigma}$.

(b) Note que $\lambda xy\left[\max(x,y)\right] = \lambda xy\left[x + (y \dot{-} x)\right]$.

(c) Note que $\lambda xy\left[x = y\right] = \lambda xy\left[1 \dot{-} ((x \dot{-} y) + (y \dot{-} x))\right]$.

(d) Note que $\lambda xy\left[x \leq y\right] = \lambda xy\left[1 \dot{-} (x \dot{-} y)\right]$.

(e) Sea $\leq$ un orden total sobre Σ. Ya que

$$\alpha = \beta \text{ sii } \#^{\leq}(\alpha) = \#^{\leq}(\beta)$$

tenemos que

$$\lambda\alpha\beta\left[\alpha = \beta\right] = \lambda xy\left[x = y\right] \circ \left[\#^{\leq} \circ p_1^{0,2}, \#^{\leq} \circ p_2^{0,2}\right]$$

lo cual nos dice que $\lambda\alpha\beta\left[\alpha = \beta\right]$ es Σ-p.r. $\blacksquare$

Operaciones logicas entre predicados Dados predicados $P : S \subseteq \omega^n \times \Sigma^{*m} \rightarrow \omega$ y $Q : S \subseteq \omega^n \times \Sigma^{*m} \rightarrow \omega$, con el mismo dominio, definamos nuevos predicados $(P \vee Q)$, $(P \wedge Q)$ y $\neg P$ de la siguiente manera

$$\begin{array}{rcl} (P \vee Q) : S & \rightarrow & \omega \\ (\vec{x}, \vec{\alpha}) & \rightarrow & \left\{ \begin{array}{ll} 1 & \text{si } P(\vec{x}, \vec{\alpha}) = 1 \text{ o } Q(\vec{x}, \vec{\alpha}) = 1 \\ 0 & \text{caso contrario} \end{array} \right. \\ (P \wedge Q) : S & \rightarrow & \omega \\ (\vec{x}, \vec{\alpha}) & \rightarrow & \left\{ \begin{array}{ll} 1 & \text{si } P(\vec{x}, \vec{\alpha}) = 1 \text{ y } Q(\vec{x}, \vec{\alpha}) = 1 \\ 0 & \text{caso contrario} \end{array} \right. \end{array}$$

$$\begin{array}{rcl} \neg P : S & \rightarrow & \omega \\ (\vec{x}, \vec{\alpha}) & \rightarrow & \left\{ \begin{array}{ll} 1 & \text{si } P(\vec{x}, \vec{\alpha}) = 0 \\ 0 & \text{si } P(\vec{x}, \vec{\alpha}) = 1 \end{array} \right. \end{array}$$

Lemma 51 *Si $P : S \subseteq \omega^n \times \Sigma^{*m} \rightarrow \omega$ y $Q : S \subseteq \omega^n \times \Sigma^{*m} \rightarrow \omega$ son predicados Σ-p.r., entonces $(P \vee Q)$, $(P \wedge Q)$ y $\neg P$ lo son tambien.*

Proof. Note que

$$\begin{array}{rcl} \neg P & = & \lambda xy\left[x \dot{-} y\right] \circ [C_1^{n,m}, P] \\ (P \wedge Q) & = & \lambda xy\left[x.y\right] \circ [P, Q] \\ (P \vee Q) & = & \neg(\neg P \wedge \neg Q). \end{array}$$

$\blacksquare$

Conjuntos Σ-recursivos primitivos Un conjunto Σ-mixto $S \subseteq \omega^n \times \Sigma^{*m}$ es llamado Σ-*recursivo primitivo* si su funcion caracteristica $\chi_S^{\omega^n \times \Sigma^{*m}}$ es Σ-p.r.. (Notese que $\chi_S^{\omega^n \times \Sigma^{*m}}$ es el predicado $\lambda\vec{x}\vec{\alpha}\left[(\vec{x}, \vec{\alpha}) \in S\right]$.)

Lemma 52 *Si $S_1, S_2 \subseteq \omega^n \times \Sigma^{*m}$ son Σ-p.r., entonces $S_1 \cup S_2$, $S_1 \cap S_2$ y $S_1 - S_2$ lo son.*

Proof. Note que

$$\begin{array}{rcl}\chi_{S_1\cup S_2}^{\omega^n\times\Sigma^{*m}} & = & (\chi_{S_1}^{\omega^n\times\Sigma^{*m}} \vee \chi_{S_2}^{\omega^n\times\Sigma^{*m}}) \\ \chi_{S_1\cap S_2}^{\omega^n\times\Sigma^{*m}} & = & (\chi_{S_1}^{\omega^n\times\Sigma^{*m}} \wedge \chi_{S_2}^{\omega^n\times\Sigma^{*m}}) \\ \chi_{S_1-S_2}^{\omega^n\times\Sigma^{*m}} & = & \lambda xy\left[x\dot{-}y\right]\circ\left[\chi_{S_1}^{\omega^n\times\Sigma^{*m}},\chi_{S_2}^{\omega^n\times\Sigma^{*m}}\right]\end{array}$$

■

Corollary 53 *Si $S\subseteq\omega^n\times\Sigma^{*m}$ es finito, entonces S es Σ-p.r..*

Proof. Si $S=\emptyset$, entonces es claro que S es Σ-p.r.. Probaremos ahora el lema para el caso en que S tiene un solo elemento. Supongamos entonces

$$S=\{(z_1,...,z_n,\gamma_1,...,\gamma_m)\}.$$

Note que $\chi_S^{\omega^n\times\Sigma^{*m}}$ es el siguiente predicado

$$\left(\chi_{\{z_1\}}^{\omega}\circ p_1^{n,m}\wedge...\wedge\chi_{\{z_n\}}^{\omega}\circ p_n^{n,m}\wedge\chi_{\{\gamma_1\}}^{\Sigma^*}\circ p_{n+1}^{n,m}\wedge...\wedge\chi_{\{\gamma_m\}}^{\Sigma^*}\circ p_{n+m}^{n,m}\right).$$

Ya que los predicados

$$\begin{array}{rcl}\chi_{\{z_i\}}^{\omega} & = & \lambda xy\left[x=y\right]\circ\left[p_1^{1,0},C_{z_i}^{1,0}\right] \\ \chi_{\{\gamma_i\}}^{\Sigma^*} & = & \lambda\alpha\beta\left[\alpha=\beta\right]\circ\left[p_1^{0,1},C_{\gamma_i}^{0,1}\right]\end{array}$$

son Σ-p.r., el Lema 51 (aplicado $(n+m)-1$ veces), implica que $\chi_S^{\omega^n\times\Sigma^{*m}}$ es Σ-p.r.. Cuando S tiene mas de un elemento, ya que entonces es la union de una cantidad finita de conjuntos de un solo elemento, se puede aplicar el Lema 52 ($|S|-1$ veces) para obtener que S es Σ-p.r.. ■

El siguiente lema caracteriza cuando un conjunto rectangular es Σ-p.r..

Lemma 54 *Supongamos $S_1,...,S_n\subseteq\omega$, $L_1,...,L_m\subseteq\Sigma^*$ son conjuntos no vacios. Entonces $S_1\times...\times S_n\times L_1\times...\times L_m$ es Σ-p.r. sii $S_1,...,S_n,L_1,...,L_m$ son Σ-p.r.*

Proof. ($\Rightarrow$) Veremos por ejemplo que L_1 es Σ-p.r.. Sea $(z_1,...,z_n,\zeta_1,...,\zeta_m)$ un elemento fijo de $S_1\times...\times S_n\times L_1\times...\times L_m$. Note que

$$\alpha\in L_1 \text{ sii } (z_1,...,z_n,\alpha,\zeta_2,...,\zeta_m)\in S_1\times...\times S_n\times L_1\times...\times L_m,$$

lo cual implica que

$$\chi_{L_1}^{\Sigma^*}=\chi_{S_1\times...\times S_n\times L_1\times...\times L_m}^{\omega^n\times\Sigma^{*m}}\circ\left[C_{z_1}^{0,1},...,C_{z_n}^{0,1},p_1^{0,1},C_{\zeta_2}^{0,1},...,C_{\zeta_m}^{0,1}\right]$$

($\Leftarrow$) Note que $\chi_{S_1\times...\times S_n\times L_1\times...\times L_m}^{\omega^n\times\Sigma^{*m}}$ es el predicado

$$\left(\chi_{S_1}^{\omega}\circ p_1^{n,m}\wedge...\wedge\chi_{S_n}^{\omega}\circ p_n^{n,m}\wedge\chi_{L_1}^{\Sigma^*}\circ p_{n+1}^{n,m}\wedge...\wedge\chi_{L_m}^{\Sigma^*}\circ p_{n+m}^{n,m}\right).$$

$\blacksquare$

Dada una funcion f y un conjunto $S \subseteq D_f$, usaremos $f|_S$ para denotar la *restriccion* de f al conjunto S, i.e. $f|_S = f \cap (S \times I_f)$. Notese que $f|_S$ es la funcion dada por

$$D_{f|_S} = S \quad \text{y} \quad f|_S(e) = f(e), \text{ para cada } e \in S$$

Lemma 55 *Sean $n, m \in \omega$ y $O \in \{\omega, \Sigma^*\}$. Supongamos $f : D_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$ es Σ-p.r.. Si $S \subseteq D_f$ es Σ-p.r., entonces $f|_S$ es Σ-p.r..*

Proof. Supongamos $O = \Sigma^*$. Entonces

$$f|_S = \lambda x\alpha \left[\alpha^x\right] \circ \left[Suc \circ Pred \circ \chi_S^{\omega^n \times \Sigma^{*m}}, f\right]$$

lo cual nos dice que $f|_S$ es Σ-p.r.. El caso $O = \omega$ es similar usando $\lambda xy \left[x^y\right]$ en lugar de $\lambda x\alpha \left[\alpha^x\right]$. $\blacksquare$

Usando el lema anterior en combinacion con el Lema 51 podemos ver que muchos predicados usuales son Σ-p.r.. Por ejemplo sea

$$P = \lambda x\alpha\beta\gamma \left[x = |\gamma| \wedge \alpha = \gamma^{Pred(|\beta|)}\right].$$

Notese que

$$D_P = \omega \times \Sigma^* \times (\Sigma^* - \{\varepsilon\}) \times \Sigma^*$$

es Σ-p.r. ya que

$$\chi_{D_P}^{\omega \times \Sigma^{*3}} = \neg \lambda\alpha\beta \left[\alpha = \beta\right] \circ \left[p_3^{1,3}, C_\varepsilon^{1,3}\right]$$

Tambien note que los predicados

$$\lambda x\alpha\beta\gamma \left[x = |\gamma|\right]$$
$$\lambda x\alpha\beta\gamma \left[\alpha = \gamma^{Pred(|\beta|)}\right]$$

son Σ-p.r. ya que pueden obtenerse componiendo funciones Σ-p.r.. O sea que P es Σ-p.r. ya que

$$P = \left(\lambda x\alpha\beta\gamma \left[x = |\gamma|\right]|_{D_P} \wedge \lambda x\alpha\beta\gamma \left[\alpha = \gamma^{Pred(|\beta|)}\right]\right).$$

@@finpagina@@

Lemma 56 *Sean $n, m \in \omega$ y $O \in \{\omega, \Sigma^*\}$. Si $f : D_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$ es Σ-p.r., entonces existe una funcion Σ-p.r. $\bar{f} : \omega^n \times \Sigma^{*m} \rightarrow O$, tal que $f = \bar{f}|_{D_f}$.*

Proof. Es facil ver por induccion en k que el enunciado se cumple para cada $f \in \mathrm{PR}_k^\Sigma$ $\blacksquare$

Ahora podemos probar el siguiente importante resultado

Proposition 57 *Un conjunto S es Σ-p.r. sii S es el dominio de alguna funcion Σ-p.r..*

Proof. Supongamos que $S \subseteq \omega^n \times \Sigma^{*m}$.

($\Rightarrow$) Note que $S = D_{Pred \circ \chi_S^{\omega^n \times \Sigma^{*m}}}$.

($\Leftarrow$) Probaremos por induccion en k que D_F es Σ-p.r., para cada $F \in \mathrm{PR}_k^\Sigma$. El caso $k = 0$ es facil. Supongamos el resultado vale para un k fijo y supongamos $F \in \mathrm{PR}_{k+1}^\Sigma$. Veremos entonces que D_F es Σ-p.r.. Hay varios casos. Consideremos primero el caso en que $F = R(f, g)$, donde

$$\begin{array}{lll} f & : & S_1 \times ... \times S_n \times L_1 \times ... \times L_m \to \Sigma^* \\ g & : & \omega \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m \times \Sigma^* \to \Sigma^*, \end{array}$$

con $S_1, ..., S_n \subseteq \omega$ y $L_1, ..., L_m \subseteq \Sigma^*$ conjuntos no vacios y $f, g \in \mathrm{PR}_k^\Sigma$. Notese que por definicion de $R(f, g)$, tenemos que

$$D_F = \omega \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m.$$

Por hipotesis inductiva tenemos que $D_f = S_1 \times ... \times S_n \times L_1 \times ... \times L_m$ es Σ-p.r., lo cual por el Lema 54 nos dice que los conjuntos $S_1, ..., S_n$, $L_1, ..., L_m$ son Σ-p.r.. Ya que ω es Σ-p.r., el Lema 54 nos dice que D_F es Σ-p.r..

Los otros casos de recursion primitiva son dejados al lector.

Supongamos ahora que $F = g \circ [g_1, ..., g_r]$ con $g, g_1, ..., g_r \in \mathrm{PR}_k^\Sigma$. Si $F = \emptyset$, entonces es claro que $D_F = \emptyset$ es Σ-p.r.. Supongamos entonces que F no es la funcion $\emptyset$. Tenemos entonces que r es de la forma $n + m$ y

$$\begin{array}{lll} g & : & D_g \subseteq \omega^n \times \Sigma^{*m} \to O \\ g_i & : & D_{g_i} \subseteq \omega^k \times \Sigma^{*l} \to \omega,\ i = 1, ..., n \\ g_i & : & D_{g_i} \subseteq \omega^k \times \Sigma^{*l} \to \Sigma^*, i = n + 1, ..., n + m \end{array}$$

con $O \in \{\omega, \Sigma^*\}$ y $k, l \in \omega$. Por Lema 56, hay funciones Σ-p.r. $\bar{g}_1, ..., \bar{g}_{n+m}$ las cuales son Σ-totales y cumplen

$$g_i = \bar{g}_i|_{D_{g_i}}, \text{ para } i = 1, ..., n + m.$$

Por hipotesis inductiva los conjuntos D_g, D_{g_i}, $i = 1, ..., n + m$, son Σ-p.r. y por lo tanto

$$S = \bigcap_{i=1}^{n+m} D_{g_i}$$

lo es. Notese que

$$\chi_{D_F}^{\omega^k \times \Sigma^{*l}} = (\chi_{D_g}^{\omega^n \times \Sigma^{*m}} \circ [\bar{g}_1, ..., \bar{g}_{n+m}] \wedge \chi_S^{\omega^k \times \Sigma^{*l}})$$

lo cual nos dice que D_F es Σ-p.r.. ■

Lema de division por casos para funciones Σ-p.r. Una observacion interesante es que si $f_i : D_{f_i} \to O$, $i = 1, ..., k$, son funciones tales que $D_{f_i} \cap D_{f_j} = \emptyset$ para $i \neq j$, entonces $f_1 \cup ... \cup f_k$ es la funcion

$$\begin{array}{rcl} D_{f_1} \cup ... \cup D_{f_k} & \to & O \\ e & \to & \left\{ \begin{array}{ll} f_1(e) & \text{si } e \in D_{f_1} \\ \vdots & \vdots \\ f_k(e) & \text{si } e \in D_{f_k} \end{array} \right. \end{array}$$

Lemma 58 *Sean $n, m \in \omega$ y $O \in \{\omega, \Sigma^*\}$. Supongamos $f_i : D_{f_i} \subseteq \omega^n \times \Sigma^{*m} \to O$, $i = 1, ..., k$, son funciones Σ-p.r. tales que $D_{f_i} \cap D_{f_j} = \emptyset$ para $i \neq j$. Entonces $f_1 \cup ... \cup f_k$ es Σ-p.r..*

Proof. Supongamos $O = \Sigma^*$ y $k = 2$. Sean

$$\bar{f}_i : \omega^n \times \Sigma^{*m} \to \Sigma^*, i = 1, 2,$$

funciones Σ-p.r. tales que $\bar{f}_i|_{D_{f_i}} = f_i$, $i = 1, 2$ (Lema 56). Por Lema 57 los conjuntos D_{f_1} y D_{f_2} son Σ-p.r. y por lo tanto lo es $D_{f_1} \cup D_{f_2}$. Ya que

$$f_1 \cup f_2 = \left(\lambda\alpha\beta\left[\alpha\beta\right] \circ \left[\lambda x\alpha\left[\alpha^x\right] \circ \left[\chi_{D_{f_1}}^{\omega^n \times \Sigma^{*m}}, \bar{f}_1 \right], \lambda x\alpha\left[\alpha^x\right] \circ \left[\chi_{D_{f_2}}^{\omega^n \times \Sigma^{*m}}, \bar{f}_2 \right] \right] \right) |_{D_{f_1} \cup D_{f_2}}$$

tenemos que $f_1 \cup f_2$ es Σ-p.r..

El caso $k > 2$ puede probarse por induccion ya que

$$f_1 \cup ... \cup f_k = (f_1 \cup ... \cup f_{k-1}) \cup f_k.$$

■

Corollary 59 *Supongamos f es una funcion Σ-mixta cuyo dominio es finito. Entonces f es Σ-p.r..*

Proof. Supongamos $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to O$, con $D_f = \{e_1, ..., e_k\}$. Por el Corolario 53, cada $\{e_i\}$ es Σ-p.r. por lo cual el Lema 55 nos dice que $C_{f(e_i)}^{n,m}|_{\{e_i\}}$ es Σ-p.r.. O sea que

$$f = C_{f(e_1)}^{n,m}|_{\{e_1\}} \cup ... \cup C_{f(e_k)}^{n,m}|_{\{e_k\}}$$

es Σ-p.r.. ■

Recordemos que dados $i \in \omega$ y $\alpha \in \Sigma^*$, definimos

$$[\alpha]_i = \left\{ \begin{array}{ll} i\text{-esimo elemento de } \alpha & \text{si } 1 \leq i \leq |\alpha| \\ \varepsilon & \text{caso contrario} \end{array} \right.$$

Lemma 60 *$\lambda i\alpha\left[[\alpha]_i\right]$ es Σ-p.r..*

Proof. Note que

$$\begin{array}{rcl} [\varepsilon]_i & = & \varepsilon \\ [\alpha a]_i & = & \left\{ \begin{array}{ll} [\alpha]_i & \text{si } i \neq |\alpha|+1 \\ a & \text{si } i = |\alpha|+1 \end{array} \right. \end{array}$$

lo cual dice que $\lambda i\alpha\left[[\alpha]_i\right] = R\left(C_\varepsilon^{1,0}, \mathcal{G}\right)$, donde $\mathcal{G}_a : \omega \times \Sigma^* \times \Sigma^* \to \Sigma^*$ es dada por

$$\mathcal{G}_a(i, \alpha, \zeta) = \left\{ \begin{array}{ll} \zeta & \text{si } i \neq |\alpha|+1 \\ a & \text{si } i = |\alpha|+1 \end{array} \right.$$

O sea que solo resta probar que cada $\mathcal{G}_a$ es Σ-p.r.. Primero note que los conjuntos

$$\begin{array}{rcl} S_1 & = & \{(i, \alpha, \zeta) \in \omega \times \Sigma^* \times \Sigma^* : i \neq |\alpha|+1\} \\ S_2 & = & \{(i, \alpha, \zeta) \in \omega \times \Sigma^* \times \Sigma^* : i = |\alpha|+1\} \end{array}$$

son Σ-p.r. ya que

$$\begin{array}{rcl} \chi_{S_1}^{\omega \times \Sigma^* \times \Sigma^*} & = & \lambda xy\left[x \neq y\right] \circ \left[p_1^{1,2}, Suc \circ \lambda\alpha\left[|\alpha|\right] \circ p_2^{1,2}\right] \\ \chi_{S_2}^{\omega \times \Sigma^* \times \Sigma^*} & = & \lambda xy\left[x = y\right] \circ \left[p_1^{1,2}, Suc \circ \lambda\alpha\left[|\alpha|\right] \circ p_2^{1,2}\right] \end{array}$$

Ya que

$$\mathcal{G}_a = p_3^{1,2}|_{S_1} \cup C_a^{1,2}|_{S_2}$$

el Lema 58 nos dice que $\mathcal{G}_a$ es Σ-p.r., para cada $a \in \Sigma$. ■

Sumatoria, productoria y concatenatoria de funciones Σ-p.r. Sea Σ un alfabeto finito. Sea $f : \omega \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m \to \omega$, con $S_1, ..., S_n \subseteq \omega$ y $L_1, ..., L_m \subseteq \Sigma^*$ no vacios. Para $x, y \in \omega$ y $(\vec{x}, \vec{\alpha}) \in S_1 \times ... \times S_n \times L_1 \times ... \times L_m$, definamos

$$\begin{array}{rcl} \sum\limits_{t=x}^{t=y} f(t, \vec{x}, \vec{\alpha}) & = & \left\{ \begin{array}{ll} 0 & \text{si } x > y \\ f(x, \vec{x}, \vec{\alpha}) + f(x+1, \vec{x}, \vec{\alpha}) + ... + f(y, \vec{x}, \vec{\alpha}) & \text{si } x \leq y \end{array} \right. \\ \prod\limits_{t=x}^{t=y} f(t, \vec{x}, \vec{\alpha}) & = & \left\{ \begin{array}{ll} 1 & \text{si } x > y \\ f(x, \vec{x}, \vec{\alpha}).f(x+1, \vec{x}, \vec{\alpha}) .... f(y, \vec{x}, \vec{\alpha}) & \text{si } x \leq y \end{array} \right. \end{array}$$

En forma similar, cuando $I_f \subseteq \Sigma^*$, definamos

$$\underset{t=x}{\overset{t=y}{\subset}} f(t, \vec{x}, \vec{\alpha}) = \left\{ \begin{array}{ll} \varepsilon & \text{si } x > y \\ f(x, \vec{x}, \vec{\alpha}) f(x+1, \vec{x}, \vec{\alpha}) .... f(y, \vec{x}, \vec{\alpha}) & \text{si } x \leq y \end{array} \right.$$

Note que, en virtud de la definicion anterior, el dominio de las funciones

$$\lambda xy\vec{x}\vec{\alpha}\left[\sum_{t=x}^{t=y} f(t, \vec{x}, \vec{\alpha})\right] \qquad \lambda xy\vec{x}\vec{\alpha}\left[\prod_{t=x}^{t=y} f(t, \vec{x}, \vec{\alpha})\right] \qquad \lambda xy\vec{x}\vec{\alpha}\left[\subset_{t=x}^{t=y} f(t, \vec{x}, \vec{\alpha})\right]$$

es $\omega \times \omega \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m$.

Lemma 61 *Sea Σ un alfabeto finito.*

(a) Si $f:\omega\times S_1\times ...\times S_n\times L_1\times ...\times L_m\rightarrow\omega$ es Σ-p.r., con $S_1,...,S_n\subseteq\omega$ y $L_1,...,L_m\subseteq\Sigma^*$ no vacios, entonces las funciones $\lambda xy\vec{x}\vec{\alpha}\left[\sum_{t=x}^{t=y}f(t,\vec{x},\vec{\alpha})\right]$ y $\lambda xy\vec{x}\vec{\alpha}\left[\prod_{t=x}^{t=y}f(t,\vec{x},\vec{\alpha})\right]$ son Σ-p.r.

(b) Si $f:\omega\times S_1\times ...\times S_n\times L_1\times ...\times L_m\rightarrow\Sigma^*$ es Σ-p.r., con $S_1,...,S_n\subseteq\omega$ y $L_1,...,L_m\subseteq\Sigma^*$ no vacios, entonces la funcion $\lambda xy\vec{x}\vec{\alpha}\left[\subset_{t=x}^{t=y}f(t,\vec{x},\vec{\alpha})\right]$ es Σ-p.r.

Proof. (a) Sea $G=\lambda tx\vec{x}\vec{\alpha}\left[\sum_{i=x}^{i=t}f(i,\vec{x},\vec{\alpha})\right]$. Ya que

$$\lambda xy\vec{x}\vec{\alpha}\left[\sum_{i=x}^{i=y}f(i,\vec{x},\vec{\alpha})\right]=G\circ\left[p_2^{n+2,m},p_1^{n+2,m},p_3^{n+2,m},...,p_{n+m+2}^{n+2,m}\right]$$

solo tenemos que probar que G es Σ-p.r.. Primero note que

$$\begin{array}{rcl}G(0,x,\vec{x},\vec{\alpha}) & = & \left\{\begin{array}{ll}0 & \text{si } x>0\\ f(0,\vec{x},\vec{\alpha}) & \text{si } x=0\end{array}\right.\\ G(t+1,x,\vec{x},\vec{\alpha}) & = & \left\{\begin{array}{ll}0 & \text{si } x>t+1\\ G(t,x,\vec{x},\vec{\alpha})+f(t+1,\vec{x},\vec{\alpha}) & \text{si } x\leq t+1\end{array}\right.\end{array}$$

O sea que si definimos

$$\begin{array}{rcl}h:\omega\times S_1\times ...\times S_n\times L_1\times ...\times L_m & \rightarrow & \omega\\ (x,\vec{x},\vec{\alpha}) & \rightarrow & \left\{\begin{array}{ll}0 & \text{si } x>0\\ f(0,\vec{x},\vec{\alpha}) & \text{si } x=0\end{array}\right.\end{array}$$

$$\begin{array}{rcl}g:\omega^3\times S_1\times ...\times S_n\times L_1\times ...\times L_m & \rightarrow & \omega\\ (A,t,x,\vec{x},\vec{\alpha}) & \rightarrow & \left\{\begin{array}{ll}0 & \text{si } x>t+1\\ A+f(t+1,\vec{x},\vec{\alpha}) & \text{si } x\leq t+1\end{array}\right.\end{array}$$

tenemos que $G=R(h,g)$. Es decir que solo nos falta probar que h y g son Σ-p.r.. Sean

$$\begin{array}{rcl}D_1 & = & \{(x,\vec{x},\vec{\alpha})\in\omega\times S_1\times ...\times S_n\times L_1\times ...\times L_m:x>0\}\\ D_2 & = & \{(x,\vec{x},\vec{\alpha})\in\omega\times S_1\times ...\times S_n\times L_1\times ...\times L_m:x=0\}\\ H_1 & = & \left\{(z,t,x,\vec{x},\vec{\alpha})\in\omega^3\times S_1\times ...\times S_n\times L_1\times ...\times L_m:x>t+1\right\}\\ H_2 & = & \left\{(z,t,x,\vec{x},\vec{\alpha})\in\omega^3\times S_1\times ...\times S_n\times L_1\times ...\times L_m:x\leq t+1\right\}.\end{array}$$

Notese que

$$\begin{array}{rcl}h & = & C_0^{n+1,m}|_{D_1}\cup\lambda x\vec{x}\vec{\alpha}\left[f(0,\vec{x},\vec{\alpha})\right]|_{D_2}\\ g & = & C_0^{n+3,m}|_{H_1}\cup\lambda Atx\vec{x}\vec{\alpha}\left[A+f(t+1,\vec{x},\vec{\alpha}))\right])|_{H_2}\end{array}$$

Ya que f es Σ-p.r. y

$$\begin{aligned}\lambda x\vec{x}\vec{\alpha}\left[f(0,\vec{x},\vec{\alpha})\right] &= f\circ\left[C_0^{n+1,m},p_2^{n+1,m},p_3^{n+1,m},...,p_{n+1+m}^{n+1,m}\right]\\ \lambda Atx\vec{x}\vec{\alpha}\left[A+f(t+1,\vec{x},\vec{\alpha})\right]) &= \lambda xy[x+y]\circ\left[p_1^{n+3,m},f\circ\left[Suc\circ p_2^{n+3,m},p_4^{n+3,m},...,p_{n+3+m}^{n+3,m}\right]\right]\end{aligned}$$

tenemos que $\lambda x\vec{x}\vec{\alpha}\left[f(0,\vec{x},\vec{\alpha})\right]$ y $\lambda Atx\vec{x}\vec{\alpha}\left[A+f(t+1,\vec{x},\vec{\alpha})\right])$ son Σ-p.r..O sea que para probar que h y g son Σ-p.r.solo nos falta ver que los conjuntos D_1,D_2,H_1,H_2 son Σ-p.r.. y aplicar luego el Lema 55. Veamos que por ejemplo H_1 lo es. Es decir debemos ver que $\chi_{H_1}^{\omega^{3+n}\times\Sigma^{*m}}$ es Σ-p.r.. Ya que f es Σ-p.r. tenemos que $D_f=\omega\times S_1\times...\times S_n\times L_1\times...\times L_m$ es Σ-p.r., lo cual por el Lema 54 nos dice que los conjuntos $S_1,...,S_n$, $L_1,...,L_m$ son Σ-p.r.. Ya que ω es Σ-p.r., el Lema 54 nos dice que $R=\omega^3\times S_1\times...\times S_n\times L_1\times...\times L_m$ es Σ-p.r.. Notese que $\chi_{H_1}^{\omega^{3+n}\times\Sigma^{*m}}=(\chi_R^{\omega^{3+n}\times\Sigma^{*m}}\wedge\lambda ztx\vec{x}\vec{\alpha}\left[x>t+1\right])$ por lo cual $\chi_{H_1}^{\omega^{3+n}\times\Sigma^{*m}}$ es Σ-p.r. ya que es la conjuncion de dos predicados Σ-p.r. ■

Veamos un ejemplo de como se puede aplicar el lema anterior. Sea $F=\lambda yx_1\left[\sum_{t=0}^{t=y}(x_1)^t\right]$. Es claro que $D_F=\omega^2$. Para ver que F es Σ-p.r. aplicaremos el lema anterior por lo cual es importante encontrar la f adecuada a la cual se le aplicara el lema. Tomemos $f=\lambda tx_1[(x_1)^t]$. Claramente f es Σ-p.r. por lo cual el lema anterior nos dice que

$$G=\lambda xyx_1\left[\sum_{t=x}^{t=y}f(t,x_1)\right]=\lambda xyx_1\left[\sum_{t=x}^{t=y}(x_1)^t\right]$$

es Σ-p.r.. Claramente G no es la funcion F pero es en algun sentido "mas amplia" que F ya que tiene una variable mas y se tiene que $F(y,x_1)=G(0,y,x_1)$, para cada $y,x_1\in\omega$. Es facil ver que

$$F=G\circ\left[C_0^{2,0},p_1^{2,0},p_2^{2,0}\right]$$

por lo cual F es Σ-p.r..

Cuantificacion acotada de predicados Σ-p.r. con dominio rectangular
Ses $P:S\times S_1\times...\times S_n\times L_1\times...\times L_m\to\omega$ un predicado, con $S,S_1,...,S_n\subseteq\omega$ y $L_1,...,L_m\subseteq\Sigma^*$ no vacios. Supongamos $\bar{S}\subseteq S$. Entonces la expresion Booleana

$$(\forall t\in\bar{S})_{t\leq x}\ P(t,\vec{x},\vec{\alpha})$$

depende de las variables $x,\vec{x},\vec{\alpha}$ y valdra 1 en una $(1+n+m)$-upla $(x,\vec{x},\vec{\alpha})$ cuando $P(t,\vec{x},\vec{\alpha})$ sea igual a 1 para cada $t\in\{u\in\bar{S}:u\leq x\}$; y 0 en caso contrario. Tenemos entonces que el dominio del predicado

$$\lambda x\vec{x}\vec{\alpha}\left[(\forall t\in\bar{S})_{t\leq x}\ P(t,\vec{x},\vec{\alpha})\right]$$

es $\omega \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m$. En forma analoga se define la forma de interpretar la expresion Booleana

$$(\exists t \in \bar{S})_{t \leq x}\ P(t, \vec{x}, \vec{\alpha})$$

Cabe destacar que

$$\lambda x\vec{x}\vec{\alpha}\left[(\exists t \in \bar{S})_{t \leq x}\ P(t, \vec{x}, \vec{\alpha})\right] = \neg\lambda x\vec{x}\vec{\alpha}\left[(\forall t \in \bar{S})_{t \leq x}\ \neg P(t, \vec{x}, \vec{\alpha})\right]$$

Tambien podemos cuantificar sobre variable alfabetica. Sea $P : S_1 \times ... \times S_n \times L_1 \times ... \times L_m \times L \to \omega$ un predicado, con $S_1, ..., S_n \subseteq \omega$ y $L, L_1, ..., L_m \subseteq \Sigma^*$ no vacios. Supongamos $\bar{L} \subseteq L$. Entonces la expresion Booleana

$$(\forall \alpha \in \bar{L})_{|\alpha| \leq x}\ P(\vec{x}, \vec{\alpha}, \alpha)$$

depende de las variables $x, \vec{x}, \vec{\alpha}$ y valdra 1 en una $(1+n+m)$-upla $(x, \vec{x}, \vec{\alpha})$ cuando $P(\vec{x}, \vec{\alpha}, \alpha)$ sea igual a 1 para cada $\alpha \in \{u \in \bar{L} : |\alpha| \leq x\}$; y 0 en caso contrario. Tenemos entonces que el dominio del predicado

$$\lambda x\vec{x}\vec{\alpha}\left[(\forall \alpha \in \bar{L})_{|\alpha| \leq x}\ P(\vec{x}, \vec{\alpha}, \alpha)\right]$$

es $\omega \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m$. En forma analoga se define la forma de interpretar la expresion Booleana

$$(\exists \alpha \in \bar{L})_{|\alpha| \leq x}\ P(\vec{x}, \vec{\alpha}, \alpha)$$

Cabe destacar que

$$\lambda x\vec{x}\vec{\alpha}\left[(\exists \alpha \in \bar{L})_{|\alpha| \leq x} P(\vec{x}, \vec{\alpha}, \alpha)\right] = \neg\lambda x\vec{x}\vec{\alpha}\left[(\forall \alpha \in \bar{L})_{|\alpha| \leq x} \neg P(\vec{x}, \vec{\alpha}, \alpha)\right]$$

Lemma 62 *Sea Σ un alfabeto finito.*

(a) Sea $P : S \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m \to \omega$ un predicado Σ-p.r., con $S, S_1, ..., S_n \subseteq \omega$ y $L_1, ..., L_m \subseteq \Sigma^*$ no vacios. Supongamos $\bar{S} \subseteq S$ es Σ-p.r.. Entonces $\lambda x\vec{x}\vec{\alpha}\left[(\forall t \in \bar{S})_{t \leq x}\ P(t, \vec{x}, \vec{\alpha})\right]$ y $\lambda x\vec{x}\vec{\alpha}\left[(\exists t \in \bar{S})_{t \leq x}\ P(t, \vec{x}, \vec{\alpha})\right]$ son predicados Σ-p.r..

(b) Sea $P : S_1 \times ... \times S_n \times L_1 \times ... \times L_m \times L \to \omega$ un predicado Σ-p.r., con $S_1, ..., S_n \subseteq \omega$ y $L, L_1, ..., L_m \subseteq \Sigma^*$ no vacios. Supongamos $\bar{L} \subseteq L$ es Σ-p.r.. Entonces $\lambda x\vec{x}\vec{\alpha}\left[(\forall \alpha \in \bar{L})_{|\alpha| \leq x}\ P(\vec{x}, \vec{\alpha}, \alpha)\right]$ y $\lambda x\vec{x}\vec{\alpha}\left[(\exists \alpha \in \bar{L})_{|\alpha| \leq x}\ P(\vec{x}, \vec{\alpha}, \alpha)\right]$ son predicados Σ-p.r..

Proof. (a) Sea

$$\bar{P} = P|_{\bar{S} \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m} \cup C_1^{1+n,m}|_{(\omega - \bar{S}) \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m}$$

Notese que $\bar{P}$ tiene dominio $\omega \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m$ y es Σ-p.r.. Ya que

$$\begin{array}{rcl}\lambda x\vec{x}\vec{\alpha}\left[(\forall t \in \bar{S})_{t\leq x} P(t,\vec{x},\vec{\alpha})\right] & = & \lambda x\vec{x}\vec{\alpha}\left[\prod_{t=0}^{t=x} \bar{P}(t,\vec{x},\vec{\alpha})\right] \\ & = & \lambda xy\vec{x}\vec{\alpha}\left[\prod_{t=x}^{t=y} \bar{P}(t,\vec{x},\vec{\alpha})\right] \circ \left[C_0^{1+n,m}, p_1^{1+n,m}, ..., p_{1+n+m}^{1+n,m}\right]\end{array}$$

el Lema 61 implica que $\lambda x\vec{x}\vec{\alpha}\left[(\forall t \in \bar{S})_{t\leq x}\ P(t,\vec{x},\vec{\alpha})\right]$ es Σ-p.r..

Ya que

$$\lambda x\vec{x}\vec{\alpha}\left[(\exists t \in \bar{S})_{t\leq x}\ P(t,\vec{x},\vec{\alpha})\right] = \neg\lambda x\vec{x}\vec{\alpha}\left[(\forall t \in \bar{S})_{t\leq x}\ \neg P(t,\vec{x},\vec{\alpha})\right]$$

tenemos que $\lambda x\vec{x}\vec{\alpha}\left[(\exists t \in \bar{S})_{t\leq x}\ P(t,\vec{x},\vec{\alpha})\right]$ es Σ-p.r.

(b) Haremos solo el caso del cuantificador $\forall$. Primero supongamos que $\Sigma = \emptyset$. Ya que $L, L_1, ..., L_m$ son no vacios, debera suceder que $L = L_1 = ... = L_m = \{\varepsilon\}$. Ya que $\bar{L} \subseteq L$, tenemos que $\bar{L} = \emptyset$ o $\bar{L} = \{\varepsilon\}$. Si $\bar{L} = \emptyset$, entonces

$$\begin{array}{rcl}\lambda x\vec{x}\vec{\alpha}\left[(\forall \alpha \in \bar{L})_{|\alpha|\leq x}\ P(\vec{x},\vec{\alpha},\alpha)\right] & = & \lambda x\vec{x}\vec{\alpha}\,[1] \\ & = & C_1^{1+n,m}\end{array}$$

por lo cual es Σ-p.r.

Si $\bar{L} = \{\varepsilon\}$, entonces

$$\begin{array}{rcl}\lambda x\vec{x}\vec{\alpha}\left[(\forall \alpha \in \bar{L})_{|\alpha|\leq x}\ P(\vec{x},\vec{\alpha},\alpha)\right] & = & \lambda x\vec{x}\vec{\alpha}\,[(P(\vec{x},\vec{\alpha},\varepsilon)] \\ & = & P \circ \left[p_2^{1+n,m}, ..., p_{1+n+m}^{1+n,m}, C_\varepsilon^{1+n,m},\right]\end{array}$$

por lo cual es Σ-p.r.

Ahora supongamos Σ es no vacio. Sea $\leq$ un orden total sobre Σ. Sea k el cardinal de Σ. Primero notese que

(*) $|\alpha| \leq x$ sii $\#^{\leq}(\alpha) \leq \sum_{\iota=1}^{i=x} k^i$, cualesquiera sean $x \in \omega$ y $\alpha \in \Sigma^*$

(queda como ejercicio probar (*). Sean

$$\begin{array}{rcl}\#^{\leq}(L) & = & \{\#^{\leq}(\alpha) : \alpha \in L\} \\ \#^{\leq}(\bar{L}) & = & \{\#^{\leq}(\alpha) : \alpha \in \bar{L}\}\end{array}$$

Notese que

$$\begin{array}{rcl}\chi^{\omega}_{\#^{\leq}(L)} & = & \chi^{\Sigma^*}_{L} \circ *^{\leq} \\ \chi^{\omega}_{\#^{\leq}(\bar{L})} & = & \chi^{\Sigma^*}_{\bar{L}} \circ *^{\leq}\end{array}$$

por lo cual $\#^{\leq}(L)$ y $\#^{\leq}(\bar{L})$ son Σ-p.r.. Sea $H = \lambda t\vec{x}\vec{\alpha}\left[P(\vec{x},\vec{\alpha},*^{\leq}(t))\right]$. Notese que

$$D_H = \#^{\leq}(L) \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m$$

y H es Σ-p.r.. O sea que por (a) tenemos que

$$\lambda x\vec{x}\vec{\alpha}\left[(\forall t\in\#^{\leq}(\bar{L}))_{t\leq x}H(t,\vec{x},\vec{\alpha})\right]=\lambda x\vec{x}\vec{\alpha}\left[(\forall t\in\#^{\leq}(\bar{L}))_{t\leq x}P(\vec{x},\vec{\alpha},*^{\leq}(t))\right]$$

es Σ-p.r.. Llamemos Q al predicado $\lambda x\vec{x}\vec{\alpha}\left[(\forall t\in\#^{\leq}(\bar{L}))_{t\leq x}P(\vec{x},\vec{\alpha},*^{\leq}(t))\right]$. Tenemos que

$$\begin{aligned}\lambda x\vec{x}\vec{\alpha}\left[(\forall\alpha\in\bar{L})_{|\alpha|\leq x}P(\vec{x},\vec{\alpha},\alpha)\right] &= \lambda x\vec{x}\vec{\alpha}\left[(\forall t\in\#^{\leq}(\bar{L}))_{t\leq\sum_{\iota=1}^{i=x}k^{i}}P(\vec{x},\vec{\alpha},*^{\leq}(t))\right]\text{ (por (*))}\\ &= Q\circ\left[\lambda x\vec{x}\vec{\alpha}\left[\sum_{\iota=1}^{i=x}k^{i}\right],p_{1}^{1+n,m},...,p_{1+n+m}^{1+n,m}\right]\end{aligned}$$

Pero $\lambda x\vec{x}\vec{\alpha}\left[\sum_{\iota=1}^{i=x}k^{i}\right]$ es Σ-p.r. (ejercicio), lo cual nos dice que $\lambda x\vec{x}\vec{\alpha}\left[(\forall\alpha\in\bar{L})_{|\alpha|\leq x}\ P(\vec{x},\vec{\alpha},\alpha)\right]$ lo es ∎

OBSERVACION: La cuantificacion no acotada no preserva la propiedad de ser Σ-p.r.. Como veremos mas adelante si elejimos bien al predicado Σ-p.r. P, obtenemos que el predicado $\lambda\vec{x}\vec{\alpha}\left[(\exists t\in\bar{S})\ P(t,\vec{x},\vec{\alpha})\right]$ no solo no es Σ-p.r. sino que tampoco es Σ-efectivamente computable (Teorema 134).

Algunos ejemplos en los cuales cuantificacion acotada se aplica naturalmente son dados a continuacion.

Lemma 63 *Sea Σ un alfabeto finito.*

(a) El predicado $\lambda xy\,[x \text{ divide } y]$ es Σ-p.r..

(b) El predicado $\lambda x\,[x \text{ es primo}]$ es Σ-p.r..

(c) El predicado $\lambda\alpha\beta\,[\alpha \text{ inicial } \beta]$ es Σ-p.r..

Proof. (a) Sea $P=\lambda tx_1x_2\,[x_2=t.x_1]$. Es claro que P es Σ-p.r.. El lema anterior nos dice que $\lambda xx_1x_2\,[(\exists t\in\omega)_{t\leq x}\ P(t,x_1,x_2)]$ es Σ-p.r.. Notese que x_1 divide x_2 si y solo si hay un $t\leq x_2$ tal que $x_2=t.x_1$. Esto nos dice que

$$\lambda x_1x_2\,[x_1 \text{ divide } x_2]=\lambda x_1x_2\,[(\exists t\in\omega)_{t\leq x_2}\ P(t,x_1,x_2)]$$

Pero

$$\lambda x_1x_2\,[(\exists t\in\omega)_{t\leq x_2}\ P(t,x_1,x_2)]=\lambda xx_1x_2\,[(\exists t\in\omega)_{t\leq x}\ P(t,x_1,x_2)]\circ\left[p_2^{2,0},p_1^{2,0},p_2^{2,0}\right]$$

por lo cual $\lambda x_1x_2\,[x_1 \text{ divide } x_2]$ es Σ-p.r.

(b) Ya que

$$x \text{ es primo sii } x>1\wedge((\forall t\in\omega)_{t\leq x}\ t=1\vee t=x\vee\neg(t \text{ divide } x))$$

podemos usar un argumento similar al de la prueba de (a).

(c) es dejado al lector. ∎

La idea fundamental subyacente en las aplicaciones anteriores es que en muchos casos de predicados obtenidos por cuantificacion a partir de otros predicados, la variable cuantificada tiene una cota natural en terminos de las otras variables y ntonces componiendo adecuadamente se lo puede presentar como un caso de cuantificacion acotada

3.2.4 Minimizacion y funciones Σ-recursivas

Tal como fue explicado anteriormente, para obtener la clase de las funciones Σ-recursivas debemos agregar un nuevo constructor a los ya definidos de composicion y recursion primitiva, a saber el constructor de *minimizacion.* Tiene dos casos.

Minimizacion de variable numerica Sea Σ un alfabeto finito y sea $P : D_P \subseteq \omega \times \omega^n \times \Sigma^{*m} \rightarrow \omega$ un predicado. Dado $(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m}$, cuando exista al menos un $t \in \omega$ tal que $P(t, \vec{x}, \vec{\alpha}) = 1$, usaremos $\min_t P(t, \vec{x}, \vec{\alpha})$ para denotar al menor de tales $t's$. Notese que la expresion $\min_t P(t, \vec{x}, \vec{\alpha})$ esta definida solo para aquellas $(n+m)$-uplas $(\vec{x}, \vec{\alpha})$ para las cuales hay al menos un t tal que se da $P(t, \vec{x}, \vec{\alpha}) = 1$. Dicho de otra forma, $\min_t P(t, \vec{x}, \vec{\alpha})$ no estara definida cuando para cada $t \in \omega$ se de que $(t, \vec{x}, \vec{\alpha})$ no pertenece a D_P o $P(t, \vec{x}, \vec{\alpha}) = 0$. Otro detalle importante a tener en cuenta es que la expresion $\min_t P(t, \vec{x}, \vec{\alpha})$ no depende de la variable t. Por ejemplo, las expresiones $\min_t P(t, \vec{x}, \vec{\alpha})$ y $\min_i P(i, \vec{x}, \vec{\alpha})$ son equivalentes en el sentido que estan definidas en las mismas $(n + m)$-uplas y cuando estan definidas asumen el mismo valor.

Definamos

$$M(P) = \lambda\vec{x}\vec{\alpha}\left[\min_t P(t, \vec{x}, \vec{\alpha})\right]$$

Notese que

$$\begin{array}{rcl} D_{M(P)} & = & \{(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m} : (\exists t \in \omega)\ P(t, \vec{x}, \vec{\alpha})\} \\ M(P)(\vec{x}, \vec{\alpha}) & = & \min_t P(t, \vec{x}, \vec{\alpha}), \text{ para cada } (\vec{x}, \vec{\alpha}) \in D_{M(P)} \end{array}$$

Diremos que $M(P)$ se obtiene por *minimizacion de variable numerica* a partir de P.

Veamos un ejemplo. Recordemos que dados $x_1, x_2 \in \omega$, con x_1 no nulo, el *cociente de dividir x_1 por x_2* se define como el maximo elemento del conjunto $\{t \in \omega : t.x_2 \leq x_1\}$. Sea

$$\begin{array}{rcl} Q : \omega \times \mathbf{N} & \rightarrow & \omega \\ (x_1, x_2) & \rightarrow & \text{cociente de dividir } x_1 \text{ por } x_2 \end{array}$$

Sea $P = \lambda t x_1 x_2 \left[x_1 < t.x_2\right]$. Notar que

$$\begin{array}{rcl} D_{M(P)} & = & \{(x_1, x_2) \in \omega^2 : (\exists t \in \omega)\ P(t, x_1, x_2) = 1\} \\ & = & \{(x_1, x_2) : (\exists t \in \omega)\ x_1 < t.x_2\} \\ & = & \omega \times \mathbf{N} \end{array}$$

Ademas si $(x_1, x_2) \in \omega \times \mathbf{N}$, es facil de probar que

$$\min_t \; x_1 < t.x_2 = Q(x_1, x_2) + 1$$

por lo que $M(P) = Suc \circ Q$. Si quisieramos encontrar un predicado P' tal que $M(P') = Q$, entonces podemos tomar $P' = \lambda t x_1 x_2 \left[x_1 < (t+1).x_2 \right]$ y con un poco de concentracion nos daremos cuenta que $M(P') = Q$. De todas maneras hay una forma mas facil de hacerlo y es tomando P' de tal forma que para cada $(x_1, x_2) \in D_Q$ se de que

$$Q(x_1, x_2) = \text{ unico } t \in \omega \text{ tal que } P'(t, x_1, x_2)$$

Por ejemplo se puede tomar $P' = \lambda t x_1 x_2 \left[x_1 \geq t.x_2 \text{ y } x_1 < (t+1).x_2 \right]$ que dicho sea de paso es justo la definicion de cociente dada en la escuela primaria. Dejamos al lector corroborar que $M(P') = Q$, para este ultimo P'.

Tal como lo vimos recien muchas veces que querramos encontrar un predicado P tal que $M(P)$ sea igual a una funcion dada F, sera mas facil encontrar un P el cual cumpla

$$F(\vec{x}, \vec{\alpha}) = \text{ unico } t \in \omega \text{ tal que } P(t, \vec{x}, \vec{\alpha})$$

es decir un predicado P que caracterice al valor que toma F.

Lemma 64 *Si $P : D_P \subseteq \omega \times \omega^n \times \Sigma^{*m} \rightarrow \omega$ es un predicado Σ-efectivamente computable y D_P es Σ-efectivamente computable, entonces la funcion $M(P)$ es Σ-efectivamente computable.*

Proof. Ejercicio ■

Lamentablemente si quitamos la hipotesis en el lema anterior de que D_P sea Σ-efectivamente computable, el lema resulta falso. Mas adelante veremos un contraejemplo basado en la tesis de Church (Proposicion 137). Por el momento el lector puede ejercitar su comprencion del tema convenciendose de que aun teniendo un procedimiento efectivo que compute a un predicado $P : D_P \subseteq \omega \times \omega^n \times \Sigma^{*m} \rightarrow \omega$, no es claro como construir un procedimiento efectivo que compute a $M(P)$.

Definicion de funcion Σ-recursiva Con este nuevo constructor de funciones estamos en condiciones de definir la clase de las funciones Σ-recursivas. Definamos los conjuntos $\mathrm{R}_0^\Sigma \subseteq \mathrm{R}_1^\Sigma \subseteq \mathrm{R}_2^\Sigma \subseteq ... \subseteq \mathrm{R}^\Sigma$ de la siguiente manera

$$\begin{array}{lcl}
\mathrm{R}_0^\Sigma & = & \mathrm{PR}_0^\Sigma \\
\mathrm{R}_{k+1}^\Sigma & = & \mathrm{R}_k^\Sigma \cup \left\{ f \circ [f_1, ..., f_n] : f, f_1, ..., f_r \in \mathrm{R}_k^\Sigma, \, r \geq 1 \right\} \cup \\
 & & \qquad \left\{ R(f, \mathcal{G}) : f \text{ y cada } \mathcal{G}_a \text{ pertenecen a } \mathrm{R}_k^\Sigma \right\} \cup \\
 & & \qquad \qquad \left\{ R(f, g) : f, g \in \mathrm{R}_k^\Sigma \right\} \cup \\
 & & \qquad \qquad \quad \left\{ M(P) : P \text{ es } \Sigma\text{-total y } P \in \mathrm{R}_k^\Sigma \right\} \\
\mathrm{R}^\Sigma & = & \bigcup_{k \geq 0} \mathrm{R}_k^\Sigma
\end{array}$$

Una funcion f es llamada Σ-*recursiva* si pertenece a R^{Σ}. Cabe destacar que aunque $M(P)$ fue definido para predicados no necesariamente Σ-totales, en la definicion de los conjuntos R_k^{Σ}, nos restringimos al caso en que P es Σ-total.

Notese que $\mathrm{PR}_k^{\Sigma} \subseteq \mathrm{R}_k^{\Sigma}$, para cada $k \in \omega$, por lo cual $\mathrm{PR}^{\Sigma} \subseteq \mathrm{R}^{\Sigma}$.

Proposition 65 *Si $f \in \mathrm{R}^{\Sigma}$, entonces f es Σ-efectivamente computable.*

Proof. Dejamos al lector la prueba por induccion en k de que si $f \in \mathrm{R}_k^{\Sigma}$, entonces f es Σ-efectivamente computable. ■

Daremos sin prueba el siguiente conceptualmente importante resultado.

Proposition 66 *Sea Σ un alfabeto finito. Entonces no toda funcion Σ-recursiva es Σ-p.r.*

Este resultado no es facil de probar. Mas adelante (Proposicion 106) veremos ejemplos naturales de funciones Σ-recursivas que no son Σ-p.r.. Otro ejemplo natural es la famosa funcion de Ackermann.

Lema de minimizacion acotada de variable numerica de predicados Σ-p.r. Aunque no siempre que $P \in \mathrm{R}^{\Sigma}$, tendremos que $M(P) \in \mathrm{R}^{\Sigma}$ (Proposicion 137), el siguiente lema nos garantiza que este es el caso cuando $P \in \mathrm{PR}^{\Sigma}$ y ademas da condiciones para que $M(P)$ sea Σ-p.r..

Lemma 67 *Sean $n, m \geq 0$. Sea $P : D_P \subseteq \omega \times \omega^n \times \Sigma^{*m} \to \omega$ un predicado Σ-p.r.. Entonces*

(a) $M(P)$ es Σ-recursiva.

(b) Si hay una funcion Σ-p.r. $f : \omega^n \times \Sigma^{*m} \to \omega$ tal que

$$M(P)(\vec{x}, \vec{\alpha}) = \min_t P(t, \vec{x}, \vec{\alpha}) \leq f(\vec{x}, \vec{\alpha}),\ \textit{para cada } (\vec{x}, \vec{\alpha}) \in D_{M(P)},$$

entonces $M(P)$ es Σ-p.r..

Proof. (a) Sea $\bar{P} = P \cup C_0^{n+1,m}|_{(\omega^{n+1} \times \Sigma^{*m}) - D_P}$. Note que $\bar{P}$ es Σ-p.r. (por que?). Veremos a continuacion que $M(P) = M(\bar{P})$. Notese que

$$\{t \in \omega : P(t, \vec{x}, \vec{\alpha}) = 1\} = \{t \in \omega : \bar{P}(t, \vec{x}, \vec{\alpha}) = 1\}$$

Esto claramente dice que $D_{M(P)} = D_{M(\bar{P})}$ y que $M(P)(\vec{x}, \vec{\alpha}) = M(\bar{P})(\vec{x}, \vec{\alpha})$, para cada $(\vec{x}, \vec{\alpha}) \in D_{M(P)}$., por lo cual $M(P) = M(\bar{P})$.

Veremos entonces que $M(\bar{P})$ es Σ-recursiva. Sea k tal que $\bar{P} \in \mathrm{PR}_k^{\Sigma}$. Ya que $\bar{P}$ es Σ-total y $\bar{P} \in \mathrm{PR}_k^{\Sigma} \subseteq \mathrm{R}_k^{\Sigma}$, tenemos que $M(\bar{P}) \in \mathrm{R}_{k+1}^{\Sigma}$ y por lo tanto $M(\bar{P}) \in \mathrm{R}^{\Sigma}$.

(b) Ya que $M(P)=M(\bar{P})$, basta con probar que $M(\bar{P})$ es Σ-p.r. Primero veremos que $D_{M(\bar{P})}$ es un conjunto Σ-p.r.. Notese que

$$\chi^{\omega^n\times\Sigma^{*m}}_{D_{M(\bar{P})}}=\lambda\vec{x}\vec{\alpha}\left[(\exists t\in\omega)_{t\leq f(\vec{x},\vec{\alpha})}\ \bar{P}(t,\vec{x},\vec{\alpha})\right]$$

lo cual nos dice que

$$\chi^{\omega^n\times\Sigma^{*m}}_{D_{M(\bar{P})}}=\lambda x\vec{x}\vec{\alpha}\left[(\exists t\in\omega)_{t\leq x}\ \bar{P}(t,\vec{x},\vec{\alpha})\right]\circ\left[f,p_1^{n,m},...,p_{n+m}^{n,m}\right]$$

Pero el Lema 62 nos dice que $\lambda x\vec{x}\vec{\alpha}\left[(\exists t\in\omega)_{t\leq x}\ \bar{P}(t,\vec{x},\vec{\alpha})\right]$ es Σ-p.r. por lo cual tenemos que $\chi^{\omega^n\times\Sigma^{*m}}_{D_{M(\bar{P})}}$ lo es.

Sea

$$P_1=\lambda t\vec{x}\vec{\alpha}\left[\bar{P}(t,\vec{x},\vec{\alpha})\wedge(\forall j\in\omega)_{j\leq t}\ j=t\vee\neg\bar{P}(j,\vec{x},\vec{\alpha})\right]$$

Note que P_1 es Σ-total. Dejamos al lector usando lemas anteriores probar que P_1 es Σ-p.r. Ademas notese que para $(\vec{x},\vec{\alpha})\in\omega^n\times\Sigma^{*m}$ se tiene que

$$P_1(t,\vec{x},\vec{\alpha})=1\text{ si y solo si }(\vec{x},\vec{\alpha})\in D_{M(\bar{P})}\text{ y }t=M(\bar{P})(\vec{x},\vec{\alpha})$$

Esto nos dice que

$$M(\bar{P})=\left(\lambda\vec{x}\vec{\alpha}\left[\prod_{t=0}^{f(\vec{x},\vec{\alpha})}t^{P_1(t,\vec{x},\vec{\alpha})}\right]\right)|_{D_{M(\bar{P})}}$$

por lo cual para probar que $M(\bar{P})$ es Σ-p.r. solo nos resta probar que

$$F=\lambda\vec{x}\vec{\alpha}\left[\prod_{t=0}^{f(\vec{x},\vec{\alpha})}t^{P_1(t,\vec{x},\vec{\alpha})}\right]$$

lo es. Pero

$$F=\lambda xy\vec{x}\vec{\alpha}\left[\prod_{t=x}^{y}t^{P_1(t,\vec{x},\vec{\alpha})}\right]\circ\left[C_0^{n,m},f,p_1^{n,m},...,p_{n+m}^{n,m}\right]$$

y por lo tanto el Lema 61 nos dice que F es Σ-p.r.. ■

OBSERVACION: No siempre que P sea Σ-p.r. tendremos que $M(P)$ lo sera. Notese que si $M(P)$ fuera Σ-p.r., cada ves que P lo sea, entonces tendriamos que $\mathrm{PR}^{\Sigma}=\mathrm{R}^{\Sigma}$ (justifique) lo cual contradiria la Proposicion 66. Mas adelante (Corolario 107) veremos un ejemplo de un predicado P el cual es Σ-p.r. pero $M(P)$ no es Σ-p.r.

El lema de minimizacion recien probado es muy util como veremos en los siguientes dos lemas.

Lemma 68 *Sea Σ un alfabeto finito. Las siguientes funciones son Σ-p.r.:*

(a) $$\begin{array}{rcl} Q:\omega\times\mathbf{N} & \rightarrow & \omega \\ (x,y) & \rightarrow & \textit{cociente de la division de } x \textit{ por } y \end{array}$$

(b) $$\begin{array}{rcl} R:\omega\times\mathbf{N} & \rightarrow & \omega \\ (x,y) & \rightarrow & \textit{resto de la division de } x \textit{ por } y \end{array}$$

(c) $$\begin{array}{rcl} pr:\mathbf{N} & \rightarrow & \omega \\ n & \rightarrow & n\textit{-esimo numero primo} \end{array}$$

Proof. (a) Ya vimos anteriormente que $Q = M(P')$, donde $P' = \lambda tx_1x_2\left[x_1 \geq t.x_2 \text{ y } x_1 < (t+1).x_2\right]$. Ya que P' es Σ-p.r. y

$$Q(x_1,x_2) \leq p_1^{2,0}(x_1,x_2), \text{ para cada } (x_1,x_2) \in \omega\times\mathbf{N}$$

(b) del Lema 67 implica que $Q \in \mathrm{PR}^{\Sigma}$.

(b) Notese que

$$R = \lambda xy\left[x \dot{-} Q(x,y).y\right]$$

y por lo tanto $R \in \mathrm{PR}^{\Sigma}$.

(c) Para ver que pr es Σ-p.r., veremos que la extension $h:\omega\rightarrow\omega$, dada por $h(0)=0$ y $h(n)=pr(n)$, $n\geq 1$, es Σ-p.r.. Luego $pr = h|_{\mathbf{N}}$ resultara Σ-p.r. por ser la restriccion de una funcion Σ-p.r. a un conjunto Σ-p.r.. Primero note que

$$\begin{array}{rcl} h(0) & = & 0 \\ h(t+1) & = & \min_i \left(i \text{ es primo} \wedge i > h(t)\right) \end{array}$$

O sea que $h = R\left(C_0^{0,0}, g\right)$, donde

$$\begin{array}{rcl} g:\omega\times\omega & \rightarrow & \omega \\ (A,t) & \rightarrow & \min_i \left(i \text{ es primo} \wedge i > A\right) \end{array}$$

Es decir que solo nos resta ver que g es Σ-p.r.. Pero notese que $g = M(P)$, donde $P = \lambda iAt\left[i \text{ es primo} \wedge i > A\right]$. Claramente P es Σ-p.r. por lo cual para poder aplicar (b) del lema anterior debemos encontrar una funcion $f:\omega\times\omega\rightarrow\omega$ tal que

$$M(P)(A,t) \leq f(A,t), \text{ para cada } (A,t)\in\omega^2$$

Es decir f debera cumplir

$$\min_i \left(i \text{ es primo} \wedge i > A\right) \leq f(A,t), \text{ para cada } (A,t)\in\omega^2$$

Definamos $f = \lambda At[A!+1]$. Debemos probar entonces que

$$\min_i \left(i \text{ es primo} \wedge i > A\right) \leq A!+1, \text{ para cada } A\in\omega$$

Sea p un primo tal que p divide a $A!+1$. Es facil ver que entonces $p > A$ ya que de lo contrario p dividiria a $A!$ lo cual nos diria que p divide a $1 = A!+1-A!$, lo cual es absurdo. Pero esto claramente nos dice que

$$\min_i \left(i \text{ es primo} \wedge i > A\right) \leq p \leq A!+1$$

O sea que (b) del Lema 67 implica que $g = M(P)$ es Σ-p.r. ■

Lemma 69 *Las funciones $\lambda xi\left[(x)_i\right]$ y $\lambda x\left[Lt(x)\right]$ son Σ-p.r.*

Proof. Note que $D_{\lambda xi[(x)_i]} = \mathbf{N} \times \mathbf{N}$. Sea

$$P = \lambda txi\left[\neg(pr(i)^{t+1} \text{ divide } x)\right]$$

Note que P es Σ-p.r. y que $D_P = \omega \times \omega \times \mathbf{N}$. Dejamos al lector la prueba de que $\lambda xi\left[(x)_i\right] = M(P)$. Ya que $(x)_i \leq x$, para todo $(x,i) \in \mathbf{N} \times \mathbf{N}$, (b) del Lema 67 implica que $\lambda xi\left[(x)_i\right]$ es Σ-p.r..

Veamos que $\lambda x\left[Lt(x)\right]$ es Σ-p.r.. Sea

$$Q = \lambda tx\left[(\forall i \in \mathbf{N})_{i \leq x}\ (i \leq t \vee (x)_i = 0)\right]$$

Notese que $D_Q = \omega \times \mathbf{N}$ y que ademas por el Lema 62 tenemos que Q es Σ-p.r. (dejamos al lector explicar como se aplica tal lema en este caso). Ademas notese que $\lambda x\left[Lt(x)\right] = M(Q)$ y que

$$Lt(x) \leq x, \text{para todo } x \in \mathbf{N}$$

lo cual por (b) del Lema 67 nos dice que $\lambda x\left[Lt(x)\right]$ es Σ-p.r.. ■

Para $x_1, ..., x_n \in \omega$, con $n \geq 1$, escribiremos $\langle x_1, ..., x_n\rangle$ en lugar de $\langle x_1, ..., x_n, 0, ...\rangle$.

Lemma 70 *Sea $n \geq 1$. La funcion $\lambda x_1...x_n\left[\langle x_1, ..., x_n\rangle\right]$ es Σ-p.r.*

Proof. Sea $f_n = \lambda x_1...x_n\left[\langle x_1, ..., x_n\rangle\right]$. Claramente f_1 es Σ-p.r.. Ademas note que para cada $n \geq 1$, tenemos

$$f_{n+1} = \lambda x_1...x_{n+1}\left[(f_n(x_1, ..., x_n)pr(n+1)^{x_{n+1}})\right].$$

O sea que podemos aplicar un argumento inductivo. ■

Minimizacion de variable alfabetica Supongamos que $\Sigma \neq \emptyset$. Sea $\leq$ un orden total sobre Σ. Recordemos que $\leq$ puede ser naturalmente extendido a un orden total sobre Σ^*. Sea $P : D_P \subseteq \omega^n \times \Sigma^{*m} \times \Sigma^* \to \omega$ un predicado. Cuando $(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m}$ es tal que existe al menos un $\alpha \in \Sigma^*$ tal que $P(\vec{x}, \vec{\alpha}, \alpha) = 1$, usaremos $\min_\alpha^\leq P(\vec{x}, \vec{\alpha}, \alpha)$ para denotar al menor $\alpha \in \Sigma^*$ tal que $P(\vec{x}, \vec{\alpha}, \alpha) = 1$. Notese que la expresion $\min_\alpha^\leq P(\vec{x}, \vec{\alpha}, \alpha)$ esta definida solo para aquellas $(n+m)$-uplas $(\vec{x}, \vec{\alpha})$ para las cuales hay al menos un α tal que se da $P(\vec{x}, \vec{\alpha}, \alpha) = 1$. Dicho de otra forma, $\min_\alpha^\leq P(\vec{x}, \vec{\alpha}, \alpha)$ no estara definida cuando para cada $\alpha \in \Sigma^*$ se de que $(\vec{x}, \vec{\alpha}, \alpha)$ no pertenece a D_P o $P(\vec{x}, \vec{\alpha}, \alpha) = 0$. Otro detalle importante a tener en cuenta es que la expresion $\min_\alpha^\leq P(\vec{x}, \vec{\alpha}, \alpha)$ no depende de la variable α. Por ejemplo, las expresiones $\min_\alpha^\leq P(\vec{x}, \vec{\alpha}, \alpha)$ y $\min_\beta^\leq P(\vec{x}, \vec{\alpha}, \beta)$ son equivalentes en el sentido que estan definidas en las mismas $(n+m)$-uplas y cuando estan definidas asumen el mismo valor.

Definamos

$$M^\leq(P) = \lambda\vec{x}\vec{\alpha}\left[\min_\alpha^\leq P(\vec{x}, \vec{\alpha}, \alpha)\right]$$

Notese que

$$\begin{array}{rcl} D_{M^{\leq}(P)} & = & \{(\vec{x},\vec{\alpha}) \in \omega^n \times \Sigma^{*m} : (\exists \alpha \in \Sigma^*)\ P(\vec{x},\vec{\alpha},\alpha)\} \\ M^{\leq}(P)(\vec{x},\vec{\alpha}) & = & \min_\alpha^{\leq} P(\vec{x},\vec{\alpha},\alpha) \text{, para cada } (\vec{x},\vec{\alpha}) \in D_{M^{\leq}(P)} \end{array}$$

Diremos que $M^{\leq}(P)$ es obtenida por *minimizacion de variable alfabetica* a partir de P.

Vemos un ejemplo. Sea $\Sigma = \{@, a, b, c, d, e\}$ y sea $\leq$ un orden total sobre Σ. Sea $Dir = \{\alpha_1 \in \Sigma^* : |\alpha_1|_@ = 1\}$ y definamos $U : Dir \to \Sigma^*$ de la siguiente manera

$$U(\alpha_1) = \text{unico } \alpha \text{ tal que } \alpha@ \text{ es tramo inicial de } \alpha_1$$

Sea

$$P = \lambda\alpha_1\alpha[\alpha_1 \in Dir \text{ y } \alpha@ \text{ es tramo inicial de } \alpha_1]$$

Tenemos que

$$\begin{array}{rcl} D_{M^{\leq}(P)} & = & \{\alpha_1 \in \Sigma^* : (\exists \alpha \in \Sigma^*)\ P(\alpha_1,\alpha)\} \\ & = & \{\alpha_1 \in \Sigma^* : \alpha_1 \in Dir \text{ y } (\exists \alpha \in \Sigma^*)\ \alpha@ \text{ es tramo inicial de } \alpha_1\} \\ & = & Dir \end{array}$$

y ademas es claro que $M^{\leq}(P)(\alpha_1) = U(\alpha_1)$, para cada $\alpha_1 \in Dir$, por lo cual $M^{\leq}(P) = U$. Intente explicar por que se utiizaron los nombres Dir y U.

Lema de minimizacion acotada de variable alfabetica de predicados Σ-p.r.

Lemma 71 *Supongamos que $\Sigma \neq \emptyset$. Sea $\leq$ un orden total sobre Σ, sean $n, m \geq 0$ y sea $P : D_P \subseteq \omega^n \times \Sigma^{*m} \times \Sigma^* \to \omega$ un predicado Σ-p.r.. Entonces*

(a) $M^{\leq}(P)$ es Σ-recursiva.

(b) Si existe una funcion Σ-p.r. $f : \omega^n \times \Sigma^{*m} \to \omega$ tal que

$$\left|M^{\leq}(P)(\vec{x},\vec{\alpha})\right| = \left|\min_\alpha^{\leq} P(\vec{x},\vec{\alpha},\alpha)\right| \leq f(\vec{x},\vec{\alpha}) \text{, para cada } (\vec{x},\vec{\alpha}) \in D_{M^{\leq}(P)},$$

entonces $M^{\leq}(P)$ es Σ-p.r..

Proof. Sea $Q = P \circ \left[p_2^{1+n,m}, ..., p_{1+n+m}^{1+n,m}, *^{\leq} \circ p_1^{1+n,m}\right]$. Note que

$$M^{\leq}(P) = *^{\leq} \circ M(Q)$$

lo cual por (a) del Lema 67 implica que $M^{\leq}(P)$ es Σ-recursiva.

Sea k el cardinal de Σ. Ya que

$$\left|*^{\leq}(M(Q)(\vec{x},\vec{\alpha}))\right| = \left|M^{\leq}(P)(\vec{x},\vec{\alpha})\right| \leq f(\vec{x},\vec{\alpha}),$$

para todo $(\vec{x},\vec{\alpha}) \in D_{M^{\leq}(P)} = D_{M(Q)}$, tenemos que

$$M(Q)(\vec{x},\vec{\alpha}) \leq \sum_{\iota=1}^{i=f(\vec{x},\vec{\alpha})} k^i, \text{ para cada } (\vec{x},\vec{\alpha}) \in D_{M(Q)}.$$

O sea que por (b) del Lema 67, $M(Q)$ es Σ-p.r. y por lo tanto $M^{\leq}(P)$ lo es. ■

En el ejemplo de recien vimos que $U = M(P)$, con $P = \lambda\alpha_1\alpha[\alpha@$ es tramo inicial de $\alpha_1]$ por lo cual, dado que P es Σ-p.r. y ademas

$$|U(\alpha_1)| \leq |\alpha_1|\,, \text{ para cada } \alpha_1 \in Dir$$

el lema anterior nos dice que U es Σ-p.r.

3.2.5 Conjuntos Σ-recursivamente enumerables

Ya que la nocion de funcion Σ-recursiva es el modelo matematico Godeliano del concepto de funcion Σ-efectivamente computable, nos podriamos preguntar entonces cual es el modelo matematico Godeliano del concepto de conjunto Σ-efectivamente enumerable. Si prestamos atencion a la definicion de conjunto Σ-efectivamente enumerable, notaremos que depende de la existencia de ciertas funciones Σ-efectivamente computables por lo cual la siguiente definicion cae de maduro:

Diremos que un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ sera llamado Σ-*recursivamente enumerable* cuando sea vacio o haya una funcion $F : \omega \rightarrow \omega^n \times \Sigma^{*m}$ tal que $I_F = S$ y $F_{(i)}$ sea Σ-recursiva, para cada $i \in \{1, ..., n+m\}$.

Deberia entonces quedar claro que si el concepto de funcion Σ-recursiva modeliza correctamente al concepto de funcion Σ-efectivamente computable, entonces el concepto de conjunto Σ-recursivamente enumerable recien definido modeliza correctamente al concepto de conjunto Σ-efectivamente enumerable.

3.2.6 Conjuntos Σ-recursivos

La version Godeliana del concepto de conjunto Σ-efectivamente computable es facil de dar: un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ sera llamado Σ-*recursivo* cuando la funcion $\chi_S^{\omega^n \times \Sigma^{*m}}$ sea Σ-recursiva.

3.2.7 Algunos resultados basicos

Muchos resultados ya probados para el caso primitivo recursivo pueden ser probados usando basicamente las mismas pruebas e ideas para el caso recursivo. Por ejemplo las pruebas de los siguientes cuatro lemas son identicas a las del caso primitivo recursivo

Lemma 72 *Si* $P : S \subseteq \omega^n \times \Sigma^{*m} \to \omega$ *y* $Q : S \subseteq \omega^n \times \Sigma^{*m} \to \omega$ *son predicados* Σ*-r., entonces* $(P \vee Q)$*,* $(P \wedge Q)$ *y* $\neg P$ *lo son tambien.*

Lemma 73 *Si* $S_1, S_2 \subseteq \omega^n \times \Sigma^{*m}$ *son* Σ*-r., entonces* $S_1 \cup S_2$*,* $S_1 \cap S_2$ *y* $S_1 - S_2$ *lo son.*

Lemma 74 *Supongamos* $S_1, ..., S_n \subseteq \omega$*,* $L_1, ..., L_m \subseteq \Sigma^*$ *son conjuntos no vacios. Entonces* $S_1 \times ... \times S_n \times L_1 \times ... \times L_m$ *es* Σ*-r. sii* $S_1, ..., S_n, L_1, ..., L_m$ *son* Σ*-r.*

Lemma 75 *Si* $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to O$ *es* Σ*-r. y* $S \subseteq D_f$ *es* Σ*-r., entonces* $f|_S$ *es* Σ*-r*

Tambien se puede probar una version del lema de division por casos para funciones Σ-recursivas con dominio Σ-recursivo, la cual generaliza el caso Σ-p.r.. La prueba es la misma que la del caso primitivo recursivo aunque al lema previo de existencia de extensiones lo probaremos en forma mas directa que para el caso primitivo recursivo. A saber:

Lemma 76 *Si* $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to O$ *es* Σ*-r. y* D_f *es* Σ*-r., entonces existe una funcion* Σ*-r.* $\bar{f} : \omega^n \times \Sigma^{*m} \to O$*, tal que* $f = \bar{f}|_{D_f}$

Proof. Si $f = \emptyset$, es facil de probar y dejado al lector. Supongamos entonces f es no vacia. Sin perdida de generalidad podemos suponer que $(0, ..., 0, \varepsilon, ..., \varepsilon) \in D_f$. Sea

$$\begin{array}{rcl} F : \omega^n \times \Sigma^{*m} & \rightarrow & \omega^n \times \Sigma^{*m} \\ (\vec{x}, \vec{\alpha}) & \rightarrow & \left\{ \begin{array}{ll} (\vec{x}, \vec{\alpha}) & \text{si } (\vec{x}, \vec{\alpha}) \in D_f \\ (0, ..., 0, \varepsilon, ..., \varepsilon) & \text{caso contrario} \end{array} \right. \end{array}$$

Ya que

$$\begin{array}{rcl} F_{(i)} & = & \lambda \vec{x} \vec{\alpha} \left[x_i . \chi_{D_f}^{\omega^n \times \Sigma^{*m}}(\vec{x}, \vec{\alpha}) \right], \text{ para } i = 1, ..., n \\ F_{(i)} & = & \lambda \vec{x} \vec{\alpha} \left[\alpha_i^{\chi_{D_f}^{\omega^n \times \Sigma^{*m}}(\vec{x}, \vec{\alpha})} \right], \text{ para } i = n+1, ..., n+m \end{array}$$

tenemos que cada $F_{(i)}$ es Σ-recursiva. Es claro que $\bar{f} = f \circ F$ cumple que $f = \bar{f}|_{D_f}$ por lo cual solo falta ver que $\bar{f}$ es Σ-recursiva. Pero esto es obvio ya que $F = \left[F_{(1)}, ..., F_{(n+m)} \right]$ ■

Lemma 77 *Supongamos $f_i : D_{f_i} \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$, $i = 1, ..., k$, son funciones Σ-recursivas tales que cada D_{f_i} es Σ-recursivo y $D_{f_i} \cap D_{f_j} = \emptyset$ para $i \neq j$. Entonces la funcion $f_1 \cup ... \cup f_k$ es Σ-recursiva.*

Proof. Completamente analoga a la del caso primitivo recursivo. ■

Lemma 78 *Si S es Σ-recursivo, entonces S es Σ-r.e.*

Proof. Supongamos $\emptyset \neq S \subseteq \omega^n \times \Sigma^{*m}$. Sea $(z_1, ..., z_n, \gamma_1, ..., \gamma_m) \in S$ fijo. Sea $\leq$ un orden total sobre Σ. Sea $G : \omega \rightarrow \omega^n \times \Sigma^{*m}$ dada por

$$G(x) = \left((x+1)_1, ..., (x+1)_n, *^{\leq}((x+1)_{n+1}), ..., *^{\leq}((x+1)_{n+m})\right)$$

Es claro que cada $G_{(i)}$ es Σ-recursiva y que $\operatorname{Im} G = \omega^n \times \Sigma^{*m}$.

Para $i = 1, ..., n$, definamos $F_i : \omega \rightarrow \omega$ de la siguiente manera

$$F_i(x) = \begin{cases} G_{(i)}(x) & \text{si } G(x) \in S \\ z_i & \text{caso contrario} \end{cases}$$

Para $i = n+1, ..., n+m$, definamos $F_i : \omega \rightarrow \Sigma^*$ de la siguiente manera

$$F_i(x) = \begin{cases} G_{(i)}(x) & \text{si } G(x) \in S \\ \gamma_i & \text{caso contrario} \end{cases}$$

Usando que S es Σ-recursivo podemos aplicar el lema anterior y ver que cada F_i es Σ-recursiva. Sea $F = [F_1, ..., F_{n+m}]$. Notese que $F_{(i)} = F_i$ para cada $i = 1, ..., n+m$. Esto nos dice que S es Σ-r.e. ya que $\operatorname{Im} F = S$. ■

Mas adelante (Lema 135) daremos un ejemplo natural de un conjunto que es Σ-r.e. pero el cual no es Σ-recursivo.

Deberia quedar claro que si el modelo de Godel es correcto, entonces todos los resultados probados dentro del paradigma filosofico de la computabilidad efectiva son ciertos una ves reenunciados de acuerdo al paradigma Godeliano. Tal como vimos arriba muchos de estos resultados se prueban en forma facil en su version recursiva. Sin envargo muchos otros requieren mas trabajo y es necesario utilizar algun paradigma mas constructivo (como el imperativo o el de Turing) para poder probarlos en su version recursiva. Por ejemplo consideremos el teorema siguiente dado en el contexto del paradigma filosofico:

Theorem 79 *Sea $S \subseteq \omega^n \times \Sigma^{*m}$. Son equivalentes*

(a) S es Σ-efectivamente computable

(b) S y $(\omega^n \times \Sigma^{*m}) - S$ son Σ-efectivamente enumerables

Se tiene que la version recursiva de (a)$\Rightarrow$(b) es probada sin problemas en el lema anterior pero para probar la version recursiva de (b)$\Rightarrow$(a), nos sera necesario utilizar el paradigma imperativo (Teorema 124). Lo mismo sucede con el lema de division por casos en su forma mas general (Lema 23) y con el teorema de caracterizacion de conjuntos Σ-efectivamente enumerables (Teorema 31), ambos cuando son enunciados en su version recursiva no son faciles de probar con las herramientas desarrolladas hasta ahora y nos sera necesario usar el paradigma imperativo para representar a los objetos recursivos involucrados. Estas pruebas estan en la Seccion 3.6 donde se compilan todos los resultados basicos (expresados en paradigma recursivo) y se obtienen algunos resultados los cuales en esta instancia todavia no se pueden probar ya que para obtenerlos es necesario hacer uso de la formalizacion matematica de ambos paradigmas el funcional y el imperativo (por ejemplo la existencia de un conjunto que es Σ-r.e. pero el cual no es Σ-recursivo).

@@finpagina@@

3.2.8 Recursion primitiva sobre valores anteriores

Dada una funcion $h:\omega\times S_1\times...\times S_n\times L_1\times...\times L_m\rightarrow\omega$, con $S_1,...,S_n\subseteq\omega$ y $L_1,...,L_m\subseteq\Sigma^*$, no vacios, definamos $h^{\downarrow}:\omega\times S_1\times...\times S_n\times L_1\times...\times L_m\rightarrow\omega$ de la siguiente manera

$$\begin{aligned} h^{\downarrow}(x,\vec{x},\vec{\alpha}) &= \langle h(0,\vec{x},\vec{\alpha}),h(1,\vec{x},\vec{\alpha}),...,h(x,\vec{x},\vec{\alpha})\rangle \\ &= \Pi_{i=0}^{x} pr(i+1)^{h(i,\vec{x},\vec{\alpha})} \end{aligned}$$

Lemma 80 *Supongamos*

$$\begin{aligned} f &: S_1\times...\times S_n\times L_1\times...\times L_m\rightarrow\omega \\ g &: \omega\times\omega\times S_1\times...\times S_n\times L_1\times...\times L_m\rightarrow\omega \\ h &: \omega\times S_1\times...\times S_n\times L_1\times...\times L_m\rightarrow\omega \end{aligned}$$

son funciones tales que

$$\begin{aligned} h(0,\vec{x},\vec{\alpha}) &= f(\vec{x},\vec{\alpha}) \\ h(x+1,\vec{x},\vec{\alpha}) &= g(h^{\downarrow}(x,\vec{x},\vec{\alpha}),x,\vec{x},\vec{\alpha}), \end{aligned}$$

para cada $x\in\omega$ y $(\vec{x},\vec{\alpha})\in S_1\times...\times S_n\times L_1\times...\times L_m$. Entonces h es Σ-r. (resp. Σ-p.r.) si f y g lo son.

Proof. Supongamos f,g son Σ-p.r.. Primero veremos que $h^{\downarrow}$ es Σ-r. (resp. Σ-p.r.). Notese que para cada $(\vec{x},\vec{\alpha})\in S_1\times...\times S_n\times L_1\times...\times L_m$ tenemos que

$$\begin{aligned} h^{\downarrow}(0,\vec{x},\vec{\alpha}) &= \langle h(0,\vec{x},\vec{\alpha})\rangle \\ &= \langle f(\vec{x},\vec{\alpha})\rangle \\ &= 2^{f(\vec{x},\vec{\alpha})} \\ h^{\downarrow}(x+1,\vec{x},\vec{\alpha}) &= h^{\downarrow}(x,\vec{x},\vec{\alpha})pr(x+2)^{h(x+1,\vec{x},\vec{\alpha})} \\ &= h^{\downarrow}(x,\vec{x},\vec{\alpha})pr(x+2)^{g(h^{\downarrow}(x,\vec{x},\vec{\alpha}),x,\vec{x},\vec{\alpha})} \end{aligned}$$

lo cual nos dice que $h^{\downarrow} = R(f_1, g_1)$ donde

$$\begin{array}{rcl} f_1 & = & \lambda \vec{x}\vec{\alpha}\left[2^{f(\vec{x},\vec{\alpha})}\right] \\ g_1 & = & \lambda A x \vec{x}\vec{\alpha}\left[Apr(x+2)^{g(A,x,\vec{x},\vec{\alpha})}\right] \end{array}$$

O sea que $h^{\downarrow}$ es Σ-r. (resp. Σ-p.r.) ya que f_1 y g_1 lo son. Finalmente notese que

$$h = \lambda ix[(x)_i] \circ \left[Suc \circ p_1^{1+n,m}, h^{\downarrow}\right]$$

lo cual nos dice que h es Σ-r. (resp. Σ-p.r.). ■

3.2.9 Independencia del alfabeto

Probaremos que los conceptos de Σ-recursividad y Σ-recursividad primitiva son en realidad independientes del alfabeto Σ, es decir que si f es una funcion la cual es Σ-mixta y Γ-mixta, entonces f es Σ-recursiva (resp. Σ-p.r.) sii f es Γ-recursiva (resp. Γ-p.r.).

Ya definimos para el caso de un alfabeto $\Sigma \neq \emptyset$ y $\leq$ un orden total sobre Σ, las funciones $\#^{\leq}$ y $*^{\leq}$. Sea $\Sigma = \emptyset$. Notese que el conjunto $\emptyset$ es un orden total sobre Σ (de hecho es el unico orden total sobre Σ). Definamos

$$\begin{array}{rcl} \#^{\emptyset} : \{0\} & \rightarrow & \{\varepsilon\} \\ 0 & \rightarrow & \varepsilon \end{array} \qquad\qquad \begin{array}{rcl} *^{\emptyset} : \{\varepsilon\} & \rightarrow & \{0\} \\ \varepsilon & \rightarrow & 0 \end{array}$$

Ya que $\Sigma^* = \{\varepsilon\}$, las funciones $\#^{\emptyset}$ y $*^{\emptyset}$ son biyecciones mutuamente inversas entre $\{0\}$ y Σ^*. Ademas notese que estas funciones son Σ-p.r..

Lemma 81 *Supongamos* $\Sigma \subseteq \Gamma$.

(a) Si $\leq$ *es un orden total sobre* Σ, *entonces las funciones* Σ*-mixtas* $*^{\leq}$ *y* $\#^{\leq}$ *son* Γ*-p.r..*

(b) Si $\leq'$ *es un orden total sobre* Γ, *entonces las funciones* Σ*-mixtas* $\#^{\leq'}|_{\Sigma^*}$ *y* $*^{\leq'}|_{\#^{\leq'}(\Sigma^*)}$ *son* Σ*-p.r..*

Proof. (a) Si $\Sigma = \emptyset$, entonces es facil ver que $*^{\leq}$ y $\#^{\leq}$ son Γ-p.r., y es dejado como ejercicio. Supongamos $\Sigma = \{a_1, ..., a_k\}$ con $k \geq 1$ y $\leq$ es dado por $a_1 < ... < a_k$. Sea $s_e^{\leq} : \Gamma^* \rightarrow \Gamma^*$ dada por

$$\begin{array}{rcl} s_e^{\leq}(\varepsilon) & = & a_1 \\ s_e^{\leq}(\alpha a_i) & = & \alpha a_{i+1}, \text{ si } i < k \\ s_e^{\leq}(\alpha a_k) & = & s_e^{\leq}(\alpha) a_1 \\ s_e^{\leq}(\alpha a) & = & \varepsilon, \text{ si } a \in \Gamma - \Sigma. \end{array}$$

Note que $s_e^{\leq}$ es Γ-p.r. y que $s_e^{\leq}|_{\Sigma^*} = s^{\leq}$. Ya que

$$\begin{array}{rcl} *^{\leq}(0) & = & \varepsilon \\ *^{\leq}(x+1) & = & s^{\leq}(*^{\leq}(x)) \end{array}$$

para cada $x \in \omega$, tenemos que

$$\begin{array}{rcl} *^{\leq}(0) & = & \varepsilon \\ *^{\leq}(x+1) & = & s_e^{\leq}(*^{\leq}(x)) \end{array}$$

Pero esto nos dice que $*^{\leq} = R(C_{\varepsilon}^{0,0}, g)$ donde

$$\begin{array}{rcl} g : \omega \times \Gamma^* & \rightarrow & \Gamma^* \\ (x, \alpha) & \rightarrow & s_e^{\leq}(\alpha) \end{array}$$

Pero es claro que g es Γ-p.r. por lo cual $*^{\leq}$ es Γ-p.r..

Para ver que $\#^{\leq} : \Sigma^* \rightarrow \omega$ es Γ-p.r., sea $\#_e^{\leq} : \Gamma^* \rightarrow \omega$ dada por

$$\begin{array}{rcl} \#_e^{\leq}(\varepsilon) & = & 0 \\ \#_e^{\leq}(\alpha a_i) & = & \#_e^{\leq}(\alpha).k + i \\ \#_e^{\leq}(\alpha a) & = & 0, \text{ si } a \in \Gamma - \Sigma. \end{array}$$

Ya que $\#_e^{\leq}$ es Γ-p.r., eso es $\#^{\leq} = \#_e^{\leq}|_{\Sigma^*}$.

(b) El caso $\Sigma = \emptyset$ es facil y queda como ejercicio. Supongamos entonces Σ es no vacio. Sea n el cardinal de Γ. Ya que

$$\begin{array}{rcl} \#^{\leq'}|_{\Sigma^*}(\varepsilon) & = & 0 \\ \#^{\leq'}|_{\Sigma^*}(\alpha a) & = & \#^{\leq'}|_{\Sigma^*}(\alpha).n + \#^{\leq'}(a), \text{ para cada } a \in \Sigma \end{array}$$

la funcion $\#^{\leq'}|_{\Sigma^*}$ es Σ-p.r.. O sea que el predicado $P = \lambda x\alpha \left[\#^{\leq'}|_{\Sigma^*}(\alpha) = x\right]$ es Σ-p.r.. Sea $\leq$ un orden total sobre Σ. Note que $*^{\leq'}|_{\#^{\leq'}(\Sigma^*)} = M^{\leq}(P)$, lo cual ya que

$$\left|*^{\leq'}|_{\#^{\leq'}(\Sigma^*)}(x)\right| \leq x$$

nos dice que $*^{\leq'}|_{\#^{\leq'}(\Sigma^*)}$ es Σ-p.r. (Lema 71). ■

Lemma 82 $\mathrm{PR}^{\emptyset} \subseteq \mathrm{PR}^{\Sigma}$ *y* $\mathrm{R}^{\emptyset} \subseteq \mathrm{R}^{\Sigma}$

Proof. Veamos que $\mathrm{R}^{\emptyset} \subseteq \mathrm{R}^{\Sigma}$. Probaremos por induccion en k que $\mathrm{R}_k^{\emptyset} \subseteq \mathrm{R}^{\Sigma}$. El caso $k = 0$ es trivial. Supongamos entonces que vale la hipotesis inductiva $\mathrm{R}_k^{\emptyset} \subseteq \mathrm{R}^{\Sigma}$ y veamos que $\mathrm{R}_{k+1}^{\emptyset} \subseteq \mathrm{R}^{\Sigma}$. Sea $F \in \mathrm{R}_{k+1}^{\emptyset} - \mathrm{R}_k^{\emptyset}$ veremos que $F \in \mathrm{R}^{\Sigma}$. Hay varios casos:

Caso $F = R(f, \mathcal{G})$, con

$$\begin{array}{rcl} f & : & S_1 \times ... \times S_n \times \emptyset^{*m} \rightarrow \emptyset^* \\ \mathcal{G}_a & : & S_1 \times ... \times S_n \times \emptyset^{*m} \times \emptyset^* \times \emptyset^* \rightarrow \emptyset^*, \text{ para cada } a \in \emptyset \end{array}$$

funciones en $\mathrm{R}_k^{\emptyset}$ y cada S_i no vacio. Por hipotesis inductiva tenemos que $f \in \mathrm{R}^{\Sigma}$. Notese que $\mathcal{G} = \emptyset$, lo cual nos dice que por definicion

$$\begin{array}{rcl} R(f,\mathcal{G}) : S_1 \times ... \times S_n \times \emptyset^{*m} \times \emptyset^* & \rightarrow & \emptyset^* \\ (\vec{x}, \varepsilon, ..., \varepsilon, \varepsilon) & \rightarrow & f(\vec{x}, \varepsilon, ..., \varepsilon) \end{array}$$

Es claro que $\omega^n \times \Sigma^{*m} \times \emptyset^*$ es un conjunto Σ-p.r. por lo cual las funciones $p_i^{n,m+1}|_{\omega^n \times \Sigma^{*m} \times \emptyset^*}$ son Σ-p.r. (aqui las $p_i^{n,m+1}$ son respecto de Σ). Ya que

$$R(f,\mathcal{G}) = f \circ \left[p_1^{n,m+1}|_{\omega^n \times \Sigma^{*m} \times \emptyset^*}, ..., p_{n+m}^{n,m+1}|_{\omega^n \times \Sigma^{*m} \times \emptyset^*} \right]$$

tenemos que F es Σ-recursiva

Caso $F = R(f,g)$, con

$$\begin{array}{rcl} f & : & S_1 \times ... \times S_n \times \emptyset^{*m} \rightarrow \emptyset^* \\ g & : & \omega \times S_1 \times ... \times S_n \times \emptyset^{*m} \times \emptyset^* \rightarrow \emptyset^* \end{array}$$

funciones en $\mathrm{R}_k^{\emptyset}$ y cada S_i no vacio. Por hipotesis inductiva tenemos que $f, g \in \mathrm{R}^{\Sigma}$. Notese que respecto de Σ, la funcion $R(f,g)$ no esta definida ya que por la forma de f, el dominio de g deberia ser $\omega \times S_1 \times ... \times S_n \times \emptyset^{*m} \times \Sigma^*$. Sea

$$\tilde{g} = g \circ \left[p_1^{1+n,m+1}, ..., p_{1+n+m}^{1+n,m+1}, C_{\varepsilon}^{1+n,m+1} \right]$$

(aqui las $p_i^{1+n,m+1}$ y $C_{\varepsilon}^{1+n,m+1}$ son respecto de Σ). Notese que $D_{\tilde{g}} = \omega \times S_1 \times ... \times S_n \times \emptyset^{*m} \times \Sigma^*$ y $\tilde{g}$ es Σ-recursiva. Ademas es facil ver que $F = Rf, \tilde{g})$ (respecto del alfabeto Σ) por lo cual F es Σ-recursiva

Caso $F = M(P)$, con $P : \omega \times \omega^n \times \emptyset^{*m} \rightarrow \omega$, un predicado en $\mathrm{R}_k^{\emptyset}$. Por hipotesis inductiva tenemos que $P \in \mathrm{R}^{\Sigma}$. Sea

$$\bar{P} = P \circ \left[p_1^{1+n,m}, ..., p_{1+n}^{1+n,m}, C_{\varepsilon}^{1+n,m}, ..., C_{\varepsilon}^{1+n,m} \right]$$

Notese que $\bar{P}$ es Σ-total y Σ-recursivo y ademas extiende a P. Sea

$$\tilde{P} = \lambda xy[x.y] \circ \left[\bar{P}, \chi_{\omega \times \omega^n \times \emptyset^{*m}}^{\omega \times \omega^n \times \Sigma^{*m}} \right]$$

Tambien $\tilde{P}$ es Σ-total y Σ-recursivo y extiende a P pero ademas fuera del dominio de P vale 0. Esto nos dice que $M(\tilde{P}) = M(P)$ por lo cual F es Σ-recursiva ya que $M(\tilde{P})$ lo es

Los otros casos de recursion primitiva son parecidos a los hechos y el caso de la composicion es trivial.

La prueba de que $\mathrm{PR}^{\emptyset} \subseteq \mathrm{PR}^{\Sigma}$ es muy similar. Se dejan los detalles como ejercicio para el lector ■

Sea Σ un alfabeto finito (puede ser vacio) y sea $\leq$ un orden total sobre Σ. Para $f : D_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow \omega$, definamos

$$f^{\#^{\leq}} = f \circ \left[p_1^{n+m,0}, ..., p_n^{n+m,0}, *^{\leq} \circ p_{n+1}^{n+m,0}, ..., *^{\leq} \circ p_{n+m}^{n+m,0} \right]$$

Similarmente, para $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to \Sigma^*$, definamos

$$f^{\#^{\leq}} = \#^{\leq} \circ f \circ \left[p_1^{n+m,0}, ..., p_n^{n+m,0}, *^{\leq} \circ p_{n+1}^{n+m,0}, ..., *^{\leq} \circ p_{n+m}^{n+m,0}\right]$$

Lemma 83 *Sea Γ un alfabeto finito y sea $\leq$ un orden total sobre Γ. Dada h una funcion Γ-mixta, son equivalentes*

(1) h es Γ-recursiva (resp. Γ-p.r.)

(2) $h^{\#^{\leq}}$ es $\emptyset$-recursiva (resp. $\emptyset$-p.r.)

Proof. (2)$\Rightarrow$(1). Supongamos $h : D_h \subseteq \omega^n \times \Gamma^{*m} \to \Gamma^*$ es tal que $h^{\#^{\leq}}$ es $\emptyset$-recursiva (resp. $\emptyset$-p.r.). Dejamos al lector chequear que

$$h = *^{\leq} \circ h^{\#^{\leq}} \circ \left[p_1^{n,m}, ..., p_n^{n,m}, \#^{\leq} \circ p_{n+1}^{n,m}, ..., \#^{\leq} \circ p_{n+m}^{n,m}\right]$$

(aqui las $p_i^{n,m}$ son respecto de Γ). Por el lema anterior tenemos que $h^{\#^{\leq}}$ es Γ-recursiva (resp. Γ-p.r.). Ya que (aun cuando $\Gamma = \emptyset$) tenemos que las funciones $*^{\leq}$ y $\#^{\leq}$ son Γ-p.r., tenemos que h es Γ-recursiva (resp. Γ-p.r.) ya que es composicion de funciones Γ-recursivas (resp. Γ-p.r.).

(1)$\Rightarrow$(2). El caso $\Gamma = \emptyset$ es trivial ya que $h^{\#^{\leq}}$ se define como composicion de funciones $\emptyset$-recursivas (resp. $\emptyset$-p.r.). Supongamos entonces que $\Gamma = \{a_1, ..., a_r\}$, con $a_1 < a_2 < ... < a_r$ y $r > 0$. Probaremos por induccion en k que

(*) Si $h \in \mathrm{R}_k^\Gamma$ (resp. $h \in \mathrm{PR}_k^\Gamma$), entonces $h^{\#^{\leq}}$ es $\emptyset$-recursiva (resp. $\emptyset$-p.r.).

El caso $k = 0$ es facil y dejado al lector. Supongamos (*) vale para un k fijo. Veremos que vale para $k + 1$. Sea $h \in \mathrm{R}_{k+1}^\Gamma$ (resp. $h \in \mathrm{PR}_{k+1}^\Gamma$). Hay varios casos

Caso 1. Supongamos $h = f \circ [f_1, ..., f_n]$, con $f, f_1, ..., f_n \in \mathrm{R}_k^\Gamma$ (resp. $f, f_1, ..., f_n \in \mathrm{PR}_k^\Gamma$). Por hipotesis inductiva tenemos que $f^{\#^{\leq}}, f_1^{\#^{\leq}}, ..., f_n^{\#^{\leq}}$ son $\emptyset$-recursivas (resp. $\emptyset$-p.r.). Ya que $h^{\#^{\leq}} = f^{\#^{\leq}} \circ \left[f_1^{\#^{\leq}}, ..., f_n^{\#^{\leq}}\right]$, tenemos que $h^{\#^{\leq}}$ es $\emptyset$-recursiva (resp. $\emptyset$-p.r.).

Caso 2. Supongamos $h = M(P)$, con $P : \omega \times \omega^n \times \Gamma^{*m} \to \omega$, un predicado en R_k^Γ. Ya que $h^{\#^{\leq}} = M(P^{\#^{\leq}})$, tenemos que $h^{\#^{\leq}}$ es $\emptyset$-recursiva.

Caso 3. Supongamos $h = R(f, \mathcal{G})$, con

$$\begin{array}{lcl} f & : & S_1 \times ... \times S_n \times L_1 \times ... \times L_m \to \Gamma^* \\ \mathcal{G}_a & : & S_1 \times ... \times S_n \times L_1 \times ... \times L_m \times \Gamma^* \times \Gamma^* \to \Gamma^*,\ a \in \Gamma \end{array}$$

funciones en R_k^Γ (resp. PR_k^Γ) y $S_1, ..., S_n \subseteq \omega$ y $L_1, ..., L_m \subseteq \Sigma^*$, no vacios. Notese que

$$\begin{array}{lcl} f^{\#^{\leq}} & : & S_1 \times ... \times S_n \times \#^{\leq}(L_1) \times ... \times \#^{\leq}(L_m) \to \omega \\ \mathcal{G}_a^{\#^{\leq}} & : & S_1 \times ... \times S_n \times \#^{\leq}(L_1) \times ... \times \#^{\leq}(L_m) \times \omega \times \omega \to \omega,\ a \in \Gamma \\ h^{\#^{\leq}} & : & S_1 \times ... \times S_n \times \#^{\leq}(L_1) \times ... \times \#^{\leq}(L_m) \times \omega \to \omega \end{array}$$

Por hipotesis inductiva tenemos que $f^{\#^{\leq}}$ y cada $\mathcal{G}_a^{\#^{\leq}}$ son $\emptyset$-recursivas (resp. $\emptyset$-p.r.). Sea

$$\begin{array}{rcl} i_0:\omega & \rightarrow & \omega \\ x & \rightarrow & \left\{ \begin{array}{ll} r & \text{si } r \text{ divide } x \\ R(x,r) & \text{caso contrario} \end{array} \right. \end{array}$$

y sea

$$B = \lambda x \left[Q(x \dot{-} i_0(x), r) \right]$$

(R y Q son definidas en el Lema 68). Note que i_0 y B son $\emptyset$-p.r. y que

$$*^{\leq}(x) = *^{\leq}(B(x))a_{i_0(x)}, \text{ para } x \geq 1$$

(ejercicio). Tambien tenemos para cada $(\vec{x}, \vec{y}, t) \in S_1 \times ... \times S_n \times \#^{\leq}(L_1) \times ... \times \#^{\leq}(L_m) \times \omega$ se da

$$\begin{array}{rcl} h^{\#^{\leq}}(\vec{x}, \vec{y}, t+1) & = & \#^{\leq}(h(\vec{x}, *^{\leq}(\vec{y}), *^{\leq}(t+1))) \\ & = & \#^{\leq}(h(\vec{x}, *^{\leq}(\vec{y}), *^{\leq}(B(t+1))a_{i_0(t+1)})) \\ & = & \#^{\leq}\left(\mathcal{G}_{a_{i_0(t+1)}}(\vec{x}, *^{\leq}(\vec{y}), *^{\leq}(B(t+1)), h(\vec{x}, *^{\leq}(\vec{y}), *^{\leq}(B(t+1)))\right) \\ & = & \#^{\leq}\left(\mathcal{G}_{a_{i_0(t+1)}}(\vec{x}, *^{\leq}(\vec{y}), *^{\leq}(B(t+1)), *^{\leq}(h^{\#^{\leq}}(\vec{x}, \vec{y}, B(t+1))))\right) \\ & = & \mathcal{G}^{\#^{\leq}}_{a_{i_0(t+1)}}(\vec{x}, \vec{y}, B(t+1), h^{\#^{\leq}}(\vec{x}, \vec{y}, B(t+1))) \end{array}$$

y ya que $B(t+1) < t+1$, tenemos que

$$\begin{array}{ll} (**) & h^{\#^{\leq}}(\vec{x}, \vec{y}, t+1) = \mathcal{G}^{\#^{\leq}}_{a_{i_0(t+1)}}(\vec{x}, \vec{y}, B(t+1), \left(\left\langle h^{\#^{\leq}}(\vec{x}, \vec{y}, 0), ..., h^{\#^{\leq}}(\vec{x}, \vec{y}, t)\right\rangle\right)_{B(t+1)+1}), \\ & \text{para cada } (\vec{x}, \vec{y}, t) \in S_1 \times ... \times S_n \times \#^{\leq}(L_1) \times ... \times \#^{\leq}(L_m) \times \omega \end{array}$$

A continuacion aplicaremos la idea del Lema 80. Sera mas claro asi ya que para aplicarlo directamente deberiamos cambiar el orden de los parametros de las funciones $h^{\#^{\leq}}$, $\mathcal{G}_{a_i}^{\#^{\leq}}$ componiendolas adecuadamente y seria muy engorroso notacionalmente.

Definamos

$$H = \lambda t \vec{x} \vec{y} \left[\left\langle h^{\#^{\leq}}(\vec{x}, \vec{y}, 0), ..., h^{\#^{\leq}}(\vec{x}, \vec{y}, t) \right\rangle \right]$$

Notar que

$$D_H = \omega \times S_1 \times ... \times S_n \times \#^{\leq}(L_1) \times ... \times \#^{\leq}(L_m)$$

Tenemos que

$$\begin{array}{rcl} H(0, \vec{x}, \vec{y}) & = & \left\langle h^{\#^{\leq}}(\vec{x}, \vec{y}, 0) \right\rangle = \left\langle f^{\#^{\leq}}(\vec{x}, \vec{y}) \right\rangle = 2^{f^{\#^{\leq}}(\vec{x}, \vec{y})} \\ H(t+1, \vec{x}, \vec{y}) & = & \left(H(t, \vec{x}, \vec{y}).pr(t+2)^{h^{\#^{\leq}}(\vec{x}, \vec{y}, t+1)} \right) \\ & = & \left(H(t, \vec{x}, \vec{y}).pr(t+2)^{\mathcal{G}^{\#^{\leq}}_{a_{i_0(t+1)}}(\vec{x}, \vec{y}, B(t+1), (H(t, \vec{x}, \vec{y}))_{B(t+1)+1})} \right) \text{ (por (**))} \end{array}$$

para cada $(t, \vec{x}, \vec{y}) \in \omega \times S_1 \times ... \times S_n \times \#^{\leq}(L_1) \times ... \times \#^{\leq}(L_m)$. O sea que si definimos

$$g : \omega \times \omega \times S_1 \times ... \times S_n \times \#^{\leq}(L_1) \times ... \times \#^{\leq}(L_m) \to \omega$$

por

$$g(A, t, \vec{x}, \vec{y}) = \begin{cases} \left(A.pr(t+2)^{\mathcal{G}_{a_1}^{\#^{\leq}}(\vec{x},\vec{y},B(t+1),(A)_{B(t+1)+1})}\right) & \text{si} \quad i_0(t+1) = 1 \\ \vdots & \vdots \\ \left(A.pr(t+2)^{\mathcal{G}_{a_r}^{\#^{\leq}}(\vec{x},\vec{y},B(t+1),(A)_{B(t+1)+1})}\right) & \text{si} \quad i_0(t+1) = r \end{cases}$$

tenemos que $H = R(\lambda x [2^x] \circ f^{\#^{\leq}}, g)$. Note que g es $\emptyset$-recursiva (resp. $\emptyset$-p.r.), ya que

$$g = \lambda At\vec{x}\vec{y} \left[f_1(A, t, \vec{x}, \vec{y})P_1(A, t, \vec{x}, \vec{y}) + ... + f_r(A, t, \vec{x}, \vec{y})P_r(A, t, \vec{x}, \vec{y})\right],$$

con

$$\begin{array}{rcl} f_i & = & \lambda At\vec{x}\vec{y} \left[\left(A.pr(t+2)^{\mathcal{G}_{a_i}^{\#^{\leq}}(\vec{x},\vec{y},B(t+1),(A)_{B(t+1)})}\right)\right] \\ P_i & = & \lambda At\vec{x}\vec{y} \left[i_0(t+1) = i\right] \end{array}$$

O sea que H es $\emptyset$-recursiva (resp. $\emptyset$-p.r.) y por lo tanto lo es

$$h^{\#^{\leq}} = \lambda \vec{x}\vec{y}t \left[(H(t, \vec{x}, \vec{y}))_{t+1}\right]$$

Los otros casos en los cuales h es obtenida por recursion primitiva son similares.
■

Ahora podemos probar el anunciado resultado de independencia.

Theorem 84 *Sean Σ y Γ alfabetos cualesquiera.*

(a) Supongamos una funcion f es Σ-mixta y Γ-mixta, entonces f es Σ-recursiva (resp. Σ-p.r.) sii f es Γ-recursiva (resp. Γ-p.r.).

(b) Supongamos un conjunto S es Σ-mixto y Γ-mixto, entonces S es Σ-recursivo (resp. Σ-r.e., Σ-p.r.) sii S es Γ-recursivo (resp. Γ-r.e., Γ-p.r.).

Proof. (a) Ya que f es $(\Sigma \cap \Gamma)$-mixta, podemos suponer sin perdida de generalidad que $\Sigma \subseteq \Gamma$ (por que?). Sea $\leq$ un orden total sobre Σ y sea $\leq'$ un orden total sobre Γ. Primero supongamos que f es Σ-recursiva (resp. Σ-p.r.). Probaremos que f es Γ-recursiva (resp. Γ-p.r.). Ya que f es Σ mixta, tenemos que $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to O$, con $O \in \{\omega, \Sigma^*\}$. Haremos el caso $O = \Sigma^*$. Ya que las funciones $\#^{\leq'}|_{\Sigma^*}$ y $*^{\leq'}|_{\#^{\leq'}(\Sigma^*)}$ son Σ-p.r. (Lema 81) y ademas

$$\begin{array}{rcl} f^{\#^{\leq'}} & = & \#^{\leq'} \circ f \circ \left[p_1^{n+m,0}, ..., p_n^{n+m,0}, *^{\leq'} \circ p_{n+1}^{n+m,0}, ..., *^{\leq'} \circ p_{n+m}^{n+m,0}\right] \\ & = & \#^{\leq'}|_{\Sigma^*} \circ f \circ \left[p_1^{n+m,0}, ..., p_n^{n+m,0}, *^{\leq'}|_{\#^{\leq'}(\Sigma^*)} \circ p_{n+1}^{n+m,0}, ..., *^{\leq'}|_{\#^{\leq'}(\Sigma^*)} \circ p_{n+m}^{n+m,0}\right] \end{array}$$

(justifique) tenemos que $f^{\#^{\leq'}}$ es Σ-recursiva (resp. Σ-p.r.). Por el lema anterior tenemos que $\left(f^{\#^{\leq'}}\right)^{\#^{\leq}}$ es $\emptyset$-recursiva (resp. $\emptyset$-p.r.), pero notese que $\left(f^{\#^{\leq'}}\right)^{\#^{\leq}} = f^{\#^{\leq'}}$ ya que $f^{\#^{\leq'}}$ es de tipo $(n+m, 0, \#)$, por lo cual tenemos que $f^{\#^{\leq'}}$ es $\emptyset$-recursiva (resp. $\emptyset$-p.r.). Pero esto por el lema anterior nos dice que f es Γ-recursiva (resp. Γ-p.r.).

Supongamos ahora que f es Γ-recursiva (resp. Γ-p.r.). Probaremos que f es Σ-recursiva (resp. Σ-p.r.). Ya que $\#^{\leq}$ y $*^{\leq}$ son Γ-p.r. (Lema 81), la funcion

$$f^{\#^{\leq}} = \#^{\leq} \circ f \circ \left[p_1^{n+m,0}, ..., p_n^{n+m,0}, *^{\leq} \circ p_{n+1}^{n+m,0}, ..., *^{\leq} \circ p_{n+m}^{n+m,0}\right]$$

es Γ-recursiva (resp. Γ-p.r.). Por el lema anterior $\left(f^{\#^{\leq}}\right)^{\#^{\leq'}}$ es $\emptyset$-recursiva (resp. $\emptyset$-p.r.). Pero notese que $\left(f^{\#^{\leq}}\right)^{\#^{\leq'}} = f^{\#^{\leq}}$ ya que $f^{\#^{\leq}}$ es de tipo $(n+m, 0, \#)$, por lo cual $f^{\#^{\leq}}$ es $\emptyset$-recursiva (resp. $\emptyset$-p.r.). Esto por el lema anterior nos dice que f es Σ-recursiva (resp. Σ-p.r.).

(b) Supongamos S es Σ-mixto y Γ-mixto. Ya que S es $(\Sigma \cap \Gamma)$-mixto, podemos suponer sin perdida de generalidad que $\Sigma \subseteq \Gamma$. Que

$$S \text{ es } \Sigma\text{-r.e. sii } S \text{ es } \Gamma\text{-r.e.}$$

sigue directo de (a). Supongamos ahora que S es Σ-recursivo. Veremos que S es Γ-recursivo. Supongamos S es de tipo (n,m) es decir $S \subseteq \omega^n \times \Sigma^{*m}$. Por definicion tenemos que $\chi_S^{\omega^n \times \Sigma^{*m}}$ es Σ-recursiva. Pero $\chi_S^{\omega^n \times \Sigma^{*m}}$ es tambien Γ-mixta, por lo cual (a) nos dice que $\chi_S^{\omega^n \times \Sigma^{*m}}$ es Γ-recursiva. Ademas es claro que el conjunto $(\omega^n \times \Gamma^{*m}) - (\omega^n \times \Sigma^{*m})$ es Γ-recursivo. Ya que

$$\chi_S^{\omega^n \times \Gamma^{*m}} = \chi_S^{\omega^n \times \Sigma^{*m}} \cup C_0^{n,m}|_{(\omega^n \times \Gamma^{*m}) - (\omega^n \times \Sigma^{*m})}$$

los Lemas 75 y 77 nos dicen que $\chi_S^{\omega^n \times \Gamma^{*m}}$ es Γ-recursiva (aqui $C_0^{n,m}$ es respecto del alfabeto Γ).

Supongamos ahora que S es Γ-recursivo. Veremos que S es Σ-recursivo. Por definicion tenemos que $\chi_S^{\omega^n \times \Gamma^{*m}}$ es Γ-recursiva. Ya que $\omega^n \times \Sigma^{*m}$ es Γ-recursivo, tenemos que $\chi_S^{\omega^n \times \Gamma^{*m}}|_{\omega^n \times \Sigma^{*m}}$ es Γ-recursiva. Por (a) tenemos que $\chi_S^{\omega^n \times \Gamma^{*m}}|_{\omega^n \times \Sigma^{*m}}$ es Σ-recursiva. Pero $\chi_S^{\omega^n \times \Sigma^{*m}} = \chi_S^{\omega^n \times \Gamma^{*m}}|_{\omega^n \times \Sigma^{*m}}$ por lo cual $\chi_S^{\omega^n \times \Sigma^{*m}}$ es Σ-recursiva, obteniendo que S es Σ-recursivo.

El caso primitivo recursivo es analogo y dejado al lector. ■

@@finpagina@@

3.3 El paradigma imperativo de Neumann: El lenguaje $\mathcal{S}^{\Sigma}$

En esta seccion daremos una modelizacion matematica del concepto de funcion Σ-efectivamente computable utilizando un lenguaje de programacion teorico el cual depende del alfabeto Σ. Lo llamaremos $\mathcal{S}^{\Sigma}$ a dicho lenguaje. Dado que fue el matematico Von Neumann quien contribuyo al desarrollo de la primera computadora de proposito general (es decir a la cual se le pueden hacer correr programas tal como a las computadoras actuales), nos referiremos a este paradigma de computabilidad efectiva como el paradigma de Von Neumann.

3.3.1 Sintaxis de $\mathcal{S}^{\Sigma}$

Necesitaremos algunas funciones basicas para poder describir la sintaxis de $\mathcal{S}^{\Sigma}$ en forma precisa. Llamaremos *numerales* a los siguientes simbolos

$$0\ 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9$$

Usaremos Num para denotar el conjunto de numerales. Notese que $Num \cap \omega = \emptyset$. Sea $S : Num^* \rightarrow Num^*$ definida de la siguiente manera

$$\begin{array}{rcl}
S(\varepsilon) & = & 1 \\
S(\alpha 0) & = & \alpha 1 \\
S(\alpha 1) & = & \alpha 2 \\
S(\alpha 2) & = & \alpha 3 \\
S(\alpha 3) & = & \alpha 4 \\
S(\alpha 4) & = & \alpha 5 \\
S(\alpha 5) & = & \alpha 6 \\
S(\alpha 6) & = & \alpha 7 \\
S(\alpha 7) & = & \alpha 8 \\
S(\alpha 8) & = & \alpha 9 \\
S(\alpha 9) & = & S(\alpha) 0
\end{array}$$

Definamos $\overline{} : \omega \rightarrow Num^*$ de la siguiente manera

$$\begin{array}{rcl}
\bar{0} & = & \varepsilon \\
\overline{n+1} & = & S(\bar{n})
\end{array}$$

Notese que para $n \in \mathbf{N}$, la palabra $\bar{n}$ es la notacion usual decimal de n. Recordemos que para $\alpha \in \Sigma^*$, definiamos

$$^{\frown}\alpha = \left\{ \begin{array}{lll} [\alpha]_2 \ldots [\alpha]_{|\alpha|} & \text{si} & |\alpha| \geq 2 \\ \varepsilon & \text{si} & |\alpha| \leq 1 \end{array} \right.$$

La sintaxis de $\mathcal{S}^{\Sigma}$ sera dada utilizando solo simbolos del alfabeto $\Sigma \cup \Sigma_p$, donde

$$\Sigma_p = Num \cup \left\{ \leftarrow, +, \dot{-}, ., \neq, ^{\frown}, \varepsilon, \mathrm{N}, \mathrm{K}, \mathrm{P}, \mathrm{L}, \mathrm{I}, \mathrm{F}, \mathrm{G}, \mathrm{O}, \mathrm{T}, \mathrm{B}, \mathrm{E}, \mathrm{S} \right\}.$$

Cabe aclarar que la palabra de longitud 0 no es un elemento de Σ_p sino que la letra griega ε que usualmente denota esta palabra, lo es. Tambien notese que en Σ_p hay simbolos que a veces representan operaciones como por ejemplo $+$ y $\dot{-}$, pero deberia quedar claro que en Σ_p estan los simbolos $+$ y $\dot{-}$ y no las operaciones que ellos denotan.

Las palabras de la forma $\mathrm{N}\bar{k}$ con $k \in \mathbf{N}$, son llamadas *variables numericas de* $\mathcal{S}^\Sigma$. Las palabras de la forma $\mathrm{P}\bar{k}$ con $k \in \mathbf{N}$, son llamadas *variables alfabeticas de* $\mathcal{S}^\Sigma$. Las palabras de la forma $\mathrm{L}\bar{k}$ con $k \in \mathbf{N}$, son llamadas *labels de* $\mathcal{S}^\Sigma$.

Una *instruccion basica de* $\mathcal{S}^\Sigma$ es una palabra de $(\Sigma \cup \Sigma_p)^*$ la cual es de alguna de las siguientes formas

$\mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{k} + 1$

$\mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{k} \dot{-} 1$

$\mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{n}$

$\mathrm{N}\bar{k} \leftarrow 0$

$\mathrm{P}\bar{k} \leftarrow \mathrm{P}\bar{k}.a$

$\mathrm{P}\bar{k} \leftarrow {}^\curvearrowright \mathrm{P}\bar{k}$

$\mathrm{P}\bar{k} \leftarrow \mathrm{P}\bar{n}$

$\mathrm{P}\bar{k} \leftarrow \varepsilon$

IF $\mathrm{N}\bar{k} \neq 0$ GOTO $\mathrm{L}\bar{n}$

IF $\mathrm{P}\bar{k}$ BEGINS a GOTO $\mathrm{L}\bar{n}$

GOTO $\mathrm{L}\bar{n}$

SKIP

donde $a \in \Sigma$ y $k, n \in \mathbf{N}$. Como puede observarse para que las instrucciones basicas sean mas lejibles usamos espacios entre ciertos simbolos. Por ejemplo, hemos escrito $\mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{k} + 1$ pero en realidad nos referimos a la palabra

$$\mathrm{N}\bar{k}{\leftarrow}\mathrm{N}\bar{k}{+}1$$

cuya longitud es $2\left|\bar{k}\right|+5$. Otro ejemplo, hemos escrito IF $\mathrm{P}\bar{k}$ BEGINS a GOTO $\mathrm{L}\bar{n}$ pero en realidad nos referiamos a la palabra $\mathrm{IFP}\bar{k}\mathrm{BEGINS}a\mathrm{GOTOL}\bar{n}$ cuya longitud es $\left|\bar{k}\right| + |\bar{n}| + 15$.

Una *instruccion de* $\mathcal{S}^\Sigma$ es ya sea una instruccion basica de $\mathcal{S}^\Sigma$ o una palabra de la forma αI, donde $\alpha \in \{\mathrm{L}\bar{n} : n \in \mathbf{N}\}$ y I es una instruccion basica de $\mathcal{S}^\Sigma$. Usaremos Ins^Σ para denotar el conjunto de todas las instrucciones de $\mathcal{S}^\Sigma$. Cuando la instruccion I es de la forma $\mathrm{L}\bar{n}J$ con J una instruccion basica, diremos que L$\bar{n}$ es el *label* de I. Damos a continuacion, a modo de ejemplo, la interpretacion intuitiva asociada a ciertas instrucciones basicas de $\mathcal{S}^{\Sigma}$:

INSTRUCCION	:	N$\bar{k}$ $\leftarrow$ N$\bar{k}\dot{-}1$
INTERPRETACION	:	Si el contenido de N$\bar{k}$ es 0 dejarlo sin modificar; en caso contrario disminuya en 1 el contenido de N$\bar{k}$
INSTRUCCION	:	N$\bar{k}$ $\leftarrow$ N$\bar{n}$
INTERPRETACION	:	Copiar en N$\bar{k}$ el contenido de N$\bar{n}$ (sin modificar el contenido de N$\bar{n}$)
INSTRUCCION	:	P$\bar{k}$ $\leftarrow^{\curvearrowright}$ P$\bar{k}$
INTERPRETACION	:	Si el contenido de P$\bar{k}$ es ε dejarlo sin modificar; en caso contrario remueva el 1er simbolo del contenido de P$\bar{k}$

INSTRUCCION	:	P$\bar{k}$ $\leftarrow$ P$\bar{k}.a$
INTERPRETACION	:	Modificar el contenido de P$\bar{k}$ agregandole el simbolo a a la derecha
INSTRUCCION	:	IF P$\bar{k}$ BEGINS a GOTO L$\bar{m}$
INTERPRETACION	:	Si el contenido de P$\bar{k}$ comiensa con a, ejecute la primer instruccion con label L$\bar{m}$; en caso contrario ejecute la siguiente instruccion

Un *programa de* $\mathcal{S}^{\Sigma}$ es una palabra de la forma

$$I_1 I_2 ... I_n$$

donde $n \geq 1$, $I_1, ..., I_n \in \mathrm{Ins}^{\Sigma}$ y ademas se cumple la siguiente propiedad, llamada *la ley de los GOTO*,

(G) Para cada $i \in \{1, ..., n\}$, si GOTOL$\bar{m}$ es un tramo final de I_i, entonces existe $j \in \{1, ..., n\}$ tal que I_j tiene label L$\bar{m}$

Usaremos Pro^{Σ} para denotar el conjunto de todos los programas de $\mathcal{S}^{\Sigma}$. Como es usual cuando escribamos un programa lo haremos linea por linea, con la finalidad de que sea mas lejible. Por ejemplo, escribiremos

$$\begin{array}{ll} \mathrm{L2} & \mathrm{N12} \leftarrow \mathrm{N12}\dot{-}1 \\ & \mathrm{P1} \leftarrow {}^{\curvearrowright}\mathrm{P1} \\ & \mathrm{IF\ N12} \neq 0 \mathrm{\ GOTO\ L2} \end{array}$$

en lugar de

$$\mathrm{L2N12}\leftarrow\mathrm{N12}\dot{-}1\mathrm{P1}\leftarrow^{\curvearrowright}\mathrm{P1IFN12}\neq 0\mathrm{GOTOL2}$$

Un importante resultado es el siguiente lema que garantiza que los programas pueden ser parseados en forma unica como concatenacion de instrucciones.

Lemma 85 *Se tiene que:*

(a) Si $I_1...I_n = J_1...J_m$*, con* $I_1, ..., I_n, J_1, ..., J_m \in \text{Ins}^\Sigma$*, entonces* $n = m$ *y* $I_j = J_j$ *para cada* $j \geq 1$*.*

(b) Si $\mathcal{P} \in \text{Pro}^\Sigma$*, entonces existe una unica sucesion de instrucciones* $I_1, ..., I_n$ *tal que* $\mathcal{P} = I_1...I_n$

Proof. (a) Supongamos I_n es un tramo final propio de J_m. Notar que entonces $n > 1$. Es facil ver que entonces ya sea $J_m = \text{L}\bar{u}I_n$ para algun $u \in \mathbf{N}$, o I_n es de la forma GOTO L$\bar{n}$ y J_m es de la forma wIF P$\bar{k}$ BEGINS a GOTO L$\bar{n}$ donde $w \in \{\text{L}\bar{n} : n \in \mathbf{N}\} \cup \{\varepsilon\}$. El segundo caso no puede darse porque entonces el anteultimo simbolo de I_{n-1} deberia ser S lo cual no sucede para ninguna instruccion. O sea que

$$I_1...I_n = J_1...J_{m-1}\text{L}\bar{u}I_n$$

lo cual dice que

(*) $I_1...I_{n-1} = J_1...J_{m-1}\text{L}\bar{u}$.

Es decir que L$\bar{u}$ es tramo final de I_{n-1} y por lo tanto GOTO L$\bar{u}$ es tramo final de I_{n-1}. Por (*), GOTO es tramo final de $J_1...J_{m-1}$, lo cual es impossible. Hemos llegado a una contradiccion lo cual nos dice que I_n no es un tramo final propio de J_m. Por simetria tenemos que $I_n = J_m$, lo cual usando un razonamiento inductivo nos dice que $n = m$ y $I_j = J_j$ para cada $j \geq 1$.

(b) Es consecuencia directa de (a). ■

(b) del lema anterior nos dice que dado un programa $\mathcal{P}$, tenemos univocamente determinados $n(\mathcal{P}) \in \mathbf{N}$ y $I_1^\mathcal{P}, ..., I_{n(\mathcal{P})}^\mathcal{P} \in \text{Ins}^\Sigma$ tales que $\mathcal{P} = I_1^\mathcal{P}...I_{n(\mathcal{P})}^\mathcal{P}$. Definamos tambien

$$I_i^\mathcal{P} = \varepsilon$$

cuando $i = 0$ o $i > n(\mathcal{P})$. Notese que las expresiones $n(\alpha)$ y I_i^α estan definidas solo cuando α es un programa (y i es un elemento de ω), es decir, cierta palabra del alfabeto $\Sigma \cup \Sigma_p$. O sea que cuando usemos notacion lambda que involucre dichas expresiones, el alfabeto respecto del cual usaremos dicha notacion sera $\Sigma \cup \Sigma_p$. Esto nos dice entonces que $\lambda\alpha[n(\alpha)]$ tiene dominio igual a $\text{Pro}^\Sigma \subseteq (\Sigma \cup \Sigma_p)^*$ y $\lambda i\alpha[I_i^\alpha]$ tiene dominio igual a $\omega \times \text{Pro}^\Sigma$. Para hacer mas sugestiva la notacion a veces escribiremos $\lambda\mathcal{P}[n(\mathcal{P})]$ y $\lambda i\mathcal{P}[I_i^\mathcal{P}]$ en lugar de $\lambda\alpha[n(\alpha)]$ y $\lambda i\alpha[I_i^\alpha]$.

Sera necesaria la funcion $Bas : \text{Ins}^\Sigma \to (\Sigma \cup \Sigma_p)^*$, dada por

$$Bas(I) = \begin{cases} J & \text{si } I \text{ es de la forma L}\bar{k}J \text{ con } J \in \text{Ins}^\Sigma \\ I & \text{caso contrario} \end{cases}$$

3.3.2 Semantica de $\mathcal{S}^{\Sigma}$

Definamos

$$\begin{aligned}\omega^{[\mathbf{N}]} &= \left\{(s_1, s_2, ...) \in \omega^{\mathbf{N}} : \text{ hay } n \in \mathbf{N} \text{ tal que } s_i = 0, \text{para } i \geq n\right\} \\ \Sigma^{*[\mathbf{N}]} &= \left\{(\sigma_1, \sigma_2, ...) \in \Sigma^{*\mathbf{N}} : \text{ hay } n \in \mathbf{N} \text{ tal que } \sigma_i = \varepsilon, \text{para } i \geq n\right\}.\end{aligned}$$

Asumiremos siempre que en una computacion via un programa de $\mathcal{S}^{\Sigma}$, todas exepto una cantidad finita de las variables numericas tienen el valor 0 y todas exepto una cantiad finita de las variables alfabeticas tienen el valor ε. Esto no quita generalidad a nuestra modelizacion del funcionamiento de los programas ya que todo programa envuelve una cantidad finita de variables.

Un *estado* es un par

$$(\vec{s}, \vec{\sigma}) = ((s_1, s_2, ...), (\sigma_1, \sigma_2, ...)) \in \omega^{[\mathbf{N}]} \times \Sigma^{*[\mathbf{N}]}.$$

Si $i \geq 1$, entonces diremos que s_i es el *contenido* o *valor* de la variable N$\bar{i}$ en el estado $(\vec{s}, \vec{\sigma})$ y σ_i es el *contenido* o *valor* de la variable P$\bar{i}$ en el estado $(\vec{s}, \vec{\sigma})$. Intuitivamente hablando, un estado es un par de infinituplas que contiene la informacion de que valores tienen alojados las distintas variables.

Imaginemos que corremos un programa $\mathcal{P}$ partiendo de un estado inicial $(\vec{s}, \vec{\sigma})$. Por supuesto la primera instruccion a realizar sera $I_1^{\mathcal{P}}$ pero, dado que $I_1^{\mathcal{P}}$ puede ser de tipo GOTO, la segunda instruccion que realizaremos puede no ser $I_2^{\mathcal{P}}$. Es decir en cada paso iremos decidiendo en funcion de la instruccion ejecutada cual es la siguiente instruccion a realizar. O sea que mientras corremos $\mathcal{P}$, en cada paso la informacion importante a tener en cuenta es, por una parte, cuales son los valores que tienen cada una de las variables y, por otra parte, cual es la instruccion que nos tocara realizar a continuacion. Esto da lugar al concepto de descripcion instantanea, a saber, un objeto matematico que describe en un instante dado de la computacion cuales son los valores de las variables y cual es la instruccion que se debe realizar en el instante siguiente. Mas formalmente una *descripcion instantanea* es una terna $(i, \vec{s}, \vec{\sigma})$ tal que $(\vec{s}, \vec{\sigma})$ es un estado e $i \in \omega$. Es decir que $\omega \times \omega^{[\mathbf{N}]} \times \Sigma^{*[\mathbf{N}]}$ es el conjunto formado por todas las descripciones instantaneas. Intuitivamente hablando, cuando $i \in \{1, ..., n(\mathcal{P})\}$, la descripcion instantanea $(i, \vec{s}, \vec{\sigma})$ nos dice que las variables estan en el estado $(\vec{s}, \vec{\sigma})$ y que la instruccion que *debemos realizar* es $I_i^{\mathcal{P}}$. Dado que sera conveniente para simplificar el tratamiento formal, nos abstraeremos un poco y cuando $i = 0$ o $i > n(\mathcal{P})$ pensaremos tambien que la descripcion instantanea $(i, \vec{s}, \vec{\sigma})$ nos dice que las variables estan en el estado $(\vec{s}, \vec{\sigma})$ y que debemos realizar $I_i^{\mathcal{P}} = \varepsilon$ (aunque por supuesto no podremos realizarla ya que no es una instruccion).

Dado un programa $\mathcal{P}$ definiremos a continuacion una funcion

$$S_{\mathcal{P}} : \omega \times \omega^{[\mathbf{N}]} \times \Sigma^{*[\mathbf{N}]} \to \omega \times \omega^{[\mathbf{N}]} \times \Sigma^{*[\mathbf{N}]}$$

la cual le asignara a una descripcion instantanea $(i, \vec{s}, \vec{\sigma})$ la *descripcion instantanea sucesora de* $(i, \vec{s}, \vec{\sigma})$ *con respecto a* $\mathcal{P}$. Cuando $i \in \{1, ..., n(\mathcal{P})\}$, intuitivamente hablando, $S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma})$ sera la descripcion instantanea que resulta luego de realizar $I_i^{\mathcal{P}}$ estando en el estado $(\vec{s},\vec{\sigma})$. Cuando $i=0$ o $i>n(\mathcal{P})$ definiremos $S_{\mathcal{P}}(i,\vec{s},\vec{\sigma})=(i,\vec{s},\vec{\sigma})$, lo cual es bastante intuitivo ya que si estamos en estado $(\vec{s},\vec{\sigma})$ y debemos realizar $I_i^{\mathcal{P}}=\varepsilon$, dado que ε no es una instruccion y por lo tanto no la podremos realizar, seguiremos en el mismo estado y teniendo que realizar $I_i^{\mathcal{P}}$.

Para darle una semantica mas unificada al concepto de descripcion instantanea sucesora debemos crear un nuevo verbo. El verbo "realizarp". Dada una actividad A, diremos que un individuo P *realizap* la actividad A, si P realiza A, en caso de que pueda hacerlo. O sea realizarp una actividad es realizarla si se puede.

Para dar otro ejemplo de este tipo de verbos, consideremos el verbo "comprarp", es decir "comprar si se puede". Un hijo le pide a su padre que le compre un determinado juguete y el padre le dice "si, hijo mio, te lo voy a comprarp". Luego el padre es despedido de su empleo y su cituacion economica hace que no le sea posible comprar dicho juguete. Sin envargo el padre no mintio ya que si bien no compro dicho juguete, él si lo comprop.

Con este verbo podemos describir intuitivamente $S_{\mathcal{P}}(i,\vec{s},\vec{\sigma})$:

$$\begin{array}{rcl} S_{\mathcal{P}}(i,\vec{s},\vec{\sigma}) & = & \text{desc inst que resulta luego de} \\ & & \text{realizarp } I_i^{\mathcal{P}}\text{, estando en estado } (\vec{s},\vec{\sigma}) \end{array}$$

Ahora si, daremos la definicion matematica de $S_{\mathcal{P}}(i,\vec{s},\vec{\sigma})$, segun se den distintos casos posibles.

Caso $i\notin\{1,...,n(\mathcal{P})\}$. Entonces $S_{\mathcal{P}}(i,\vec{s},\vec{\sigma})=(i,\vec{s},\vec{\sigma})$

Caso $Bas(I_i^{\mathcal{P}})=\mathrm{N}\bar{k}\leftarrow\mathrm{N}\bar{k}\dot{-}1$. Entonces

$$S_{\mathcal{P}}(i,\vec{s},\vec{\sigma})=(i+1,(s_1,...,s_{k-1},s_k\dot{-}1,s_{k+1},...),\vec{\sigma})$$

Caso $Bas(I_i^{\mathcal{P}})=\mathrm{N}\bar{k}\leftarrow\mathrm{N}\bar{k}+1$. Entonces

$$S_{\mathcal{P}}(i,\vec{s},\vec{\sigma})=(i+1,(s_1,...,s_{k-1},s_k+1,s_{k+1},...),\vec{\sigma})$$

Caso $Bas(I_i^{\mathcal{P}})=\mathrm{N}\bar{k}\leftarrow\mathrm{N}\bar{n}$. Entonces

$$S_{\mathcal{P}}(i,\vec{s},\vec{\sigma})=(i+1,(s_1,...,s_{k-1},s_n,s_{k+1},...),\vec{\sigma})$$

Caso $Bas(I_i^{\mathcal{P}})=\mathrm{N}\bar{k}\leftarrow 0$. Entonces

$$S_{\mathcal{P}}(i,\vec{s},\vec{\sigma})=(i+1,(s_1,...,s_{k-1},0,s_{k+1},...),\vec{\sigma})$$

Caso $Bas(I_i^{\mathcal{P}})=\mathrm{IF\ N}\bar{k}\neq 0\ \mathrm{GOTO\ L}\bar{m}$. Entonces tenemos dos subcasos.

Subcaso a. El valor de $\mathrm{N}\bar{k}$ en $(\vec{s},\vec{\sigma})$ es 0. Entonces

$$S_{\mathcal{P}}(i,\vec{s},\vec{\sigma})=(i+1,\vec{s},\vec{\sigma})$$

Subcaso b. El valor de $\mathrm{N}\bar{k}$ en $(\vec{s},\vec{\sigma})$ es no nulo. Entonces

$$S_{\mathcal{P}}(i,\vec{s},\vec{\sigma})=(\min\{l: I_l^{\mathcal{P}} \text{ tiene label L}\bar{m}\},\vec{s},\vec{\sigma})$$

Caso $Bas(I_i^{\mathcal{P}}) = \mathrm{P}\bar{k} \leftarrow {}^{\frown}\mathrm{P}\bar{k}$. Entonces

$$S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (i + 1, \vec{s}, (\sigma_1, ..., \sigma_{k-1}, {}^{\frown}\sigma_k, \sigma_{k+1}, ...))$$

Caso $Bas(I_i^{\mathcal{P}}) = \mathrm{P}\bar{k} \leftarrow \mathrm{P}\bar{k}.a$. Entonces

$$S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (i + 1, \vec{s}, (\sigma_1, ..., \sigma_{k-1}, \sigma_k a, \sigma_{k+1}, ...))$$

Caso $Bas(I_i^{\mathcal{P}}) = \mathrm{P}\bar{k} \leftarrow \mathrm{P}\bar{n}$. Entonces

$$S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (i + 1, \vec{s}, (\sigma_1, ..., \sigma_{k-1}, \sigma_n, \sigma_{k+1}, ...))$$

Caso $Bas(I_i^{\mathcal{P}}) = \mathrm{P}\bar{k} \leftarrow \varepsilon$. Entonces

$$S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (i + 1, \vec{s}, (\sigma_1, ..., \sigma_{k-1}, \varepsilon, \sigma_{k+1}, ...))$$

Caso $Bas(I_i^{\mathcal{P}}) = \mathrm{IF\ P}\bar{k}\ \mathrm{BEGINS}\ a\ \mathrm{GOTO\ L}\bar{m}$. Entonces tenemos dos subcasos.

Subcaso a. El valor de $\mathrm{P}\bar{k}$ en $(\vec{s}, \vec{\sigma})$ comiensa con a. Entonces

$$S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (\min\{l : I_l^{\mathcal{P}} \text{ tiene label L}\bar{m}\}, \vec{s}, \vec{\sigma})$$

Subcaso b. El valor de $\mathrm{P}\bar{k}$ en $(\vec{s}, \vec{\sigma})$ no comiensa con a. Entonces

$$S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (i + 1, \vec{s}, \vec{\sigma})$$

Caso $Bas(I_i^{\mathcal{P}}) = \mathrm{GOTO\ L}\bar{m}$. Entonces

$$S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (\min\{l : I_l^{\mathcal{P}} \text{ tiene label L}\bar{m}\}, \vec{s}, \vec{\sigma})$$

Caso $Bas(I_i^{\mathcal{P}}) = \mathrm{SKIP}$. Entonces

$$S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (i + 1, \vec{s}, \vec{\sigma})$$

La computacion partiendo de un estado Dado un programa $\mathcal{P}$ y un estado $(\vec{s}, \vec{\sigma})$ a la infinitupla

$$((1, \vec{s}, \vec{\sigma}), S_{\mathcal{P}}(1, \vec{s}, \vec{\sigma}), S_{\mathcal{P}}(S_{\mathcal{P}}(1, \vec{s}, \vec{\sigma})), S_{\mathcal{P}}(S_{\mathcal{P}}(S_{\mathcal{P}}(1, \vec{s}, \vec{\sigma}))), ...)$$

la llamaremos la *computacion de $\mathcal{P}$ partiendo del estado* $(\vec{s}, \vec{\sigma})$. Diremos que

$$\overbrace{S_{\mathcal{P}}(...S_{\mathcal{P}}(S_{\mathcal{P}}}^{t \text{ veces}}(1, \vec{s}, \vec{\sigma}))...)$$

es la *descripcion instantanea obtenida luego de t pasos, partiendo del estado* $(\vec{s}, \vec{\sigma})$. Si

$$\overbrace{S_{\mathcal{P}}(...S_{\mathcal{P}}(S_{\mathcal{P}}}^{t \text{ veces}}(1, \vec{s}, \vec{\sigma}))...) = (j, \vec{u}, \vec{\eta})$$

diremos que $(\vec{u}, \vec{\eta})$ es el *estado obtenido luego de t pasos, partiendo del estado* $(\vec{s}, \vec{\sigma})$.

Es claro que en la infinitupla de mas arriba esta toda la informacion de la "corrida" del programa $\mathcal{P}$ cuando partimos del estado $(\vec{s}, \vec{\sigma})$. Veamos un ejemplo. Sea $\Sigma = \{\blacktriangle, \#\}$ y sea $\mathcal{P}$ el siguiente programa

$$\begin{array}{ll} \mathrm{L3} & \mathrm{N4} \leftarrow \mathrm{N4}+1 \\ & \mathrm{P1} \leftarrow {}^{\frown}\mathrm{P1} \\ & \mathrm{IF\ P1\ BEGINS\ } \blacktriangle \mathrm{\ GOTO\ L3} \\ & \mathrm{P3} \leftarrow \mathrm{P3}.\# \end{array}$$

Supongamos que tomamos $(\vec{s}, \vec{\sigma})$ igual al estado

$$((2,1,0,5,3,0,0,0,...),(\#\blacktriangle\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...))$$

Tendremos entonces que la computacion de $\mathcal{P}$ partiendo del estado $(\vec{s}, \vec{\sigma})$ es la siguiente sucesion (de arriba hacia abajo) de descripciones instantaneas:

$$\begin{gathered}
(1,(2,1,0,5,3,0,0,0,...),(\#\blacktriangle\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{realizando } I_1^{\mathcal{P}} = \mathrm{N4} \leftarrow \mathrm{N4}+1 \text{ obtenemos} \\
(2,(2,1,0,6,3,0,0,0,...),(\#\blacktriangle\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{realizando } I_2^{\mathcal{P}} = \mathrm{P1} \leftarrow {}^{\frown}\mathrm{P1} \text{ obtenemos} \\
(3,(2,1,0,6,3,0,0,0,...),(\blacktriangle\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{realizando } I_3^{\mathcal{P}} = \mathrm{IF\ P1\ BEGINS\ } \blacktriangle \mathrm{\ GOTO\ L3} \text{ obtenemos} \\
(1,(2,1,0,6,3,0,0,0,...),(\blacktriangle\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{realizando } I_1^{\mathcal{P}} = \mathrm{N4} \leftarrow \mathrm{N4}+1 \text{ obtenemos} \\
(2,(2,1,0,7,3,0,0,0,...),(\blacktriangle\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{realizando } I_2^{\mathcal{P}} = \mathrm{P1} \leftarrow {}^{\frown}\mathrm{P1} \text{ obtenemos} \\
(3,(2,1,0,7,3,0,0,0,...),(\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{realizando } I_3^{\mathcal{P}} = \mathrm{IF\ P1\ BEGINS\ } \blacktriangle \mathrm{\ GOTO\ L3} \text{ obtenemos} \\
(4,(2,1,0,7,3,0,0,0,...),(\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{realizando } I_4^{\mathcal{P}} = \mathrm{P3} \leftarrow \mathrm{P3}.\# \text{ obtenemos} \\
(5,(2,1,0,7,3,0,0,0,...),(\#\#,\varepsilon,\blacktriangle\blacktriangle\#,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{intentando realizar } I_5^{\mathcal{P}} = \varepsilon \text{ obtenemos} \\
(5,(2,1,0,7,3,0,0,0,...),(\#\#,\varepsilon,\blacktriangle\blacktriangle\#,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{intentando realizar } I_5^{\mathcal{P}} = \varepsilon \text{ obtenemos} \\
(5,(2,1,0,7,3,0,0,0,...),(\#\#,\varepsilon,\blacktriangle\blacktriangle\#,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{intentando realizar } I_5^{\mathcal{P}} = \varepsilon \text{ obtenemos} \\
(5,(2,1,0,7,3,0,0,0,...),(\#\#,\varepsilon,\blacktriangle\blacktriangle\#,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\vdots
\end{gathered}$$

Notese que en este caso es natural decir que el programa $\mathcal{P}$ se detiene, partiendo del estado inicial dado ya que llega a un punto en el que queda intentando realizar $I^{\mathcal{P}}_{n(\mathcal{P})+1}$ lo cual no es una instruccion. Veamos un ejemplo de no detencion. Sea $\mathcal{Q}$ el siguiente programa

$$\begin{array}{ll} \mathrm{L3} & \mathrm{N4} \leftarrow \mathrm{N4}+1 \\ & \mathrm{IF\ P1\ BEGINS\ } \blacktriangle \mathrm{\ GOTO\ L3} \end{array}$$

Supongamos que tomamos $(\vec{s},\vec{\sigma})$ igual al estado

$$((2,1,0,5,3,0,0,0,...),(\blacktriangle\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...))$$

Tendremos entonces que la computacion de $\mathcal{Q}$ partiendo del estado $(\vec{s},\vec{\sigma})$ es la siguiente sucesion (de arriba hacia abajo) de descripciones instantaneas:

$$\begin{array}{c}
(1,(2,1,0,5,3,0,0,0,...),(\blacktriangle\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{realizando } I_1^{\mathcal{P}} = \mathrm{N4} \leftarrow \mathrm{N4}+1 \text{ obtenemos} \\
(2,(2,1,0,6,3,0,0,0,...),(\blacktriangle\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{realizando } I_2^{\mathcal{P}} = \mathrm{IF\ P1\ BEGINS\ } \blacktriangle \mathrm{\ GOTO\ L3} \text{ obtenemos} \\
(1,(2,1,0,6,3,0,0,0,...),(\blacktriangle\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{realizando } I_1^{\mathcal{P}} = \mathrm{N4} \leftarrow \mathrm{N4}+1 \text{ obtenemos} \\
(2,(2,1,0,7,3,0,0,0,...),(\blacktriangle\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{realizando } I_2^{\mathcal{P}} = \mathrm{IF\ P1\ BEGINS\ } \blacktriangle \mathrm{\ GOTO\ L3} \text{ obtenemos} \\
(1,(2,1,0,7,3,0,0,0,...),(\blacktriangle\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{realizando } I_1^{\mathcal{P}} = \mathrm{N4} \leftarrow \mathrm{N4}+1 \text{ obtenemos} \\
(2,(2,1,0,8,3,0,0,0,...),(\blacktriangle\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{realizando } I_2^{\mathcal{P}} = \mathrm{IF\ P1\ BEGINS\ } \blacktriangle \mathrm{\ GOTO\ L3} \text{ obtenemos} \\
(1,(2,1,0,8,3,0,0,0,...),(\blacktriangle\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{realizando } I_1^{\mathcal{P}} = \mathrm{N4} \leftarrow \mathrm{N4}+1 \text{ obtenemos} \\
(2,(2,1,0,9,3,0,0,0,...),(\blacktriangle\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\text{realizando } I_2^{\mathcal{P}} = \mathrm{IF\ P1\ BEGINS\ } \blacktriangle \mathrm{\ GOTO\ L3} \text{ obtenemos} \\
(1,(2,1,0,9,3,0,0,0,...),(\blacktriangle\#\#,\varepsilon,\blacktriangle\blacktriangle,\#\blacktriangle,\#,\varepsilon,\varepsilon,\varepsilon,...)) \\
\vdots
\end{array}$$

Notese que en este caso, es claro que el programa $\mathcal{Q}$ no se detiene partiendo del estado inicial dado ya que sigue indefinidamente realizando instrucciones.

Definicion matematica de detencion Ahora definiremos matematicamente el concepto de detencion. Cuando la primer coordenada de

$$\overbrace{S_{\mathcal{P}}(...S_{\mathcal{P}}(S_{\mathcal{P}}}^{t \text{ veces}}(1,\vec{s},\vec{\sigma}))...)$$

sea igual a $n(\mathcal{P})+1$, diremos que $\mathcal{P}$ *se detiene (luego de t pasos), partiendo desde el estado* $(\vec{s},\vec{\sigma})$. Si ninguna de las primeras coordenadas en la computacion

$$((1,\vec{s},\vec{\sigma}),S_{\mathcal{P}}(1,\vec{s},\vec{\sigma}),S_{\mathcal{P}}(S_{\mathcal{P}}(1,\vec{s},\vec{\sigma})),S_{\mathcal{P}}(S_{\mathcal{P}}(S_{\mathcal{P}}(1,\vec{s},\vec{\sigma}))),...)$$

es igual a $n(\mathcal{P})+1$, diremos que $\mathcal{P}$ *no se detiene partiendo del estado* $(\vec{s},\vec{\sigma})$.

Cabe destacar que en los conceptos antes definidos por "1 paso" entendemos "realizarp una instrucion", donde tal como se lo explico antes "realizarp" significa "realizar si se puede". Otra observacion importante es que los programas de $\mathcal{S}^{\Sigma}$ tienen una sola manera de detenerse, i.e. siempre que se detienen lo hacen habiendo realizado la ultima de sus instrucciones e intentando realizar la instruccion siguiente a su ultima instruccion

3.3.3 Funciones Σ-computables

Ahora que hemos definido matematicamente la semantica de $\mathcal{S}^{\Sigma}$ estamos en condiciones de definir el concepto de funcion Σ-computable, el cual sera una modelizacion matematica del concepto de funcion Σ-efectivamente computable. Intuitivamente hablando una funcion sera Σ-computable cuando haya un programa que la compute. Para precisar este concepto nos sera util la siguiente notacion. Dados $x_1,...,x_n \in \omega$ y $\alpha_1,...,\alpha_m \in \Sigma^*$, con $n,m \in \omega$, usaremos

$$\|x_1,...,x_n,\alpha_1,...,\alpha_m\|$$

para denotar el estado

$$((x_1,...,x_n,0,...),(\alpha_1,...,\alpha_m,\varepsilon,...))$$

Esta notacion requiere aclarar un poco como debe interpretarse en los casos limite, es decir cuando alguno de los numeros n,m es igual a 0. Notese que por ejemplo

$$\|x\| = ((x,0,...),(\varepsilon,...))$$

(es el caso $n=1$ y $m=0$). Tambien

$$\|\alpha\| = ((0,...),(\alpha,\varepsilon,...))$$

(es el caso $n=0$ y $m=1$). En el caso $n=m=0$ pensaremos que $x_1,...,x_n,\alpha_1,...,\alpha_m$ se transforma en $\Diamond$ por lo que se obtiene

$$\|\Diamond\| = ((0,...),(\varepsilon,...))$$

Ademas es claro que

$$\|x_1,...,x_n,\alpha_1,...,\alpha_m\| = \left\|x_1,...,x_n,\overbrace{0,...,0}^{i},\alpha_1,...,\alpha_m,\overbrace{\varepsilon,...,\varepsilon}^{j}\right\|$$

cualesquiera sean $i,j \in \omega$.

Dado $\mathcal{P} \in \text{Pro}^{\Sigma}$, definamos para cada par $n, m \geq 0$, la funcion $\Psi_{\mathcal{P}}^{n,m,\#}$ de la siguiente manera:

$$D_{\Psi_{\mathcal{P}}^{n,m,\#}} = \{(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m} : \mathcal{P} \text{ termina, partiendo del estado } \|x_1, ..., x_n, \alpha_1, ..., \alpha_m\|\}$$

$$\Psi_{\mathcal{P}}^{n,m,\#}(\vec{x}, \vec{\alpha}) = \text{valor de N1 en el estado obtenido cuando } \mathcal{P} \text{ termina, partiendo de } \|x_1, ..., x_n, \alpha_1, ..., \alpha_m\|$$

Analogamente definamos la funcion $\Psi_{\mathcal{P}}^{n,m,*}$ de la siguiente manera:

$$D_{\Psi_{\mathcal{P}}^{n,m,*}} = \{(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m} : \mathcal{P} \text{ termina, partiendo del estado } \|x_1, ..., x_n, \alpha_1, ..., \alpha_m\|\}$$

$$\Psi_{\mathcal{P}}^{n,m,*}(\vec{x}, \vec{\alpha}) = \text{valor de P1 en el estado obtenido cuando } \mathcal{P} \text{ termina, partiendo de } \|x_1, ..., x_n, \alpha_1, ..., \alpha_m\|$$

Ahora si daremos la definicion precisa de funcion Σ-computable. Una funcion Σ-mixta $f : S \subseteq \omega^n \times \Sigma^{*m} \to \omega$ sera llamada Σ-*computable* si hay un programa $\mathcal{P}$ tal que $f = \Psi_{\mathcal{P}}^{n,m,\#}$. En tal caso diremos que la funcion f es *computada* por $\mathcal{P}$. Analogamente una funcion Σ-mixta $f : S \subseteq \omega^n \times \Sigma^{*m} \to \Sigma^*$ sera llamada Σ-*computable* si hay un programa $\mathcal{P}$ tal que $f = \Psi_{\mathcal{P}}^{n,m,*}$. En tal caso diremos que la funcion f es *computada* por $\mathcal{P}$.

Algunos ejemplos:

E_1 El programa

```
L2  IF N1 ≠ 0 GOTO L1
    GOTO L2
L1  N1 ← N1∸1
```

computa la funcion $Pred$. Note que este programa tambien computa las funciones $Pred \circ p_1^{n,m}$, para $n \geq 1$ y $m \geq 0$.

E_2 Sea $\Sigma = \{\clubsuit, \triangle\}$. El programa

```
L3  IF P2 BEGINS ♣ GOTO L1
    IF P2 BEGINS △ GOTO L2
    GOTO L4
L1  P2 ← ↷P2
    P1 ← P1♣
    GOTO L3
L2  P2 ← ↷P2
    P1 ← P1△
    GOTO L3
L4  SKIP
```

computa la funcion $\lambda\alpha\beta\,[\alpha\beta]$.

Por supuesto para que el concepto de funcion Σ-computable tenga chance de ser una modelizacion adecuada del concepto de funcion Σ-efectivamente computable, tiene que ser cierto el siguiente resultado.

Proposition 86 *Si f es Σ-computable, entonces f es Σ-efectivamente computable.*

Proof. Supongamos por ejemplo que $f : S \subseteq \omega^n \times \Sigma^{*m} \to \omega$ es computada por $\mathcal{P} \in \mathrm{Pro}^{\Sigma}$. Un procedimiento efectivo que compute a f puede consistir de realizar las sucesivas instrucciones de $\mathcal{P}$ (partiendo de $\|x_1, ..., x_n, \alpha_1, ..., \alpha_m\|$) y eventualmente terminar en caso de que nos toque realizar la instruccion $n(\mathcal{P})+1$, y dar como salida el contenido de la variable N1. Daremos a continuacion una descripcion mas detallada de dicho procedimiento.

Fijemos primero un numero natural k que sea mayor que $\max\{n, m\}$ y tal que toda variable que ocurre en $\mathcal{P}$ este en la lista $\mathrm{N}1, ..., \mathrm{N}\bar{k}, \mathrm{P}1, ..., \mathrm{P}\bar{k}$. Sea $\mathbb{P}$ el siguiente procedimiento efectivo:

- Conjunto de datos de entrada de $\mathbb{P}$ igual a $\omega^n \times \Sigma^{*m}$
- Conjunto de datos de salida de $\mathbb{P}$ contenido en ω
- Funcionamiento:

Etapa 1

Asignar los siguientes valores a las variables $I, X_1, ..., X_k, A_1, ..., A_k$:

$$\begin{array}{cc}
\multicolumn{2}{c}{I \leftarrow 1} \\
X_1 \leftarrow x_1 & A_1 \leftarrow \alpha_1 \\
\vdots & \vdots \\
X_n \leftarrow x_n & A_m \leftarrow \alpha_m \\
X_{n+1} \leftarrow 0 & A_{m+1} \leftarrow \varepsilon \\
\vdots & \vdots \\
X_k \leftarrow 0 & A_k \leftarrow \varepsilon
\end{array}$$

Etapa 2

Asignar:

$I \leftarrow$ 1er coordenada de $S_{\mathcal{P}}(I, (X_1, ..., X_k, 0, ...), (A_1, ..., A_k, \varepsilon, ...))$

Para $i = 1, ..., k$:

$X_i \leftarrow i$-esima coordenada de la segunda coordenada de $S_{\mathcal{P}}(I, (X_1, ..., X_k, 0, ...), (A_1, ..., A_k, \varepsilon, ...))$

$A_i \leftarrow i$-esima coordenada de la tercer coordenada de $S_{\mathcal{P}}(I, (X_1, ..., X_k, 0, ...), (A_1, ..., A_k, \varepsilon, ...))$

Etapa 3

Si $I = n(\mathcal{P}) + 1$, entonces dar X_1 como salida y detenerse. En caso contrario ir a Etapa 2.

Se deja al lector corroborar que $\mathbb{P}$ es efectivo. ■

Sin envargo nuestro modelo imperativo de funcion Σ-efectivamente computable todavia podria no ser correcto ya que podria pasar que haya una funcion Σ-mixta que sea computada por un procedimiento efectivo pero que no exista un programa de $\mathcal{S}^{\Sigma}$ que la compute. En otras palabras el modelo imperativo o Neumanniano podria ser incompleto. Por supuesto este no es el caso y los desarrollos que veremos mas adelante nos convenceran de que el paradigma imperativo es completo.

3.3.4 Macros

Supongamos que estamos escribiendo un programa $\mathcal{P}$ de $\mathcal{S}^{\Sigma}$ con el objeto de que realice cierta tarea. Supongamos ademas que nos vendria muy bien para nuestros propositos poder usar una instruccion

$$\mathrm{N5} \leftarrow \mathrm{N16} + \mathrm{N3}$$

la cual por supuesto al correr el programa, deberia producir el efecto de dejar en la variable N5 la suma de los contenidos de las variables N16 y N3, sin modificar el contenido de las variables distintas a N5. Lamentablemente no tenemos en $\mathcal{S}^{\Sigma}$ este tipo de instruccion pero podriamos reemplazarla por el siguiente programa

$$\begin{array}{ll}
 & \mathrm{N1111} \leftarrow \mathrm{N16} \\
 & \mathrm{N2222} \leftarrow \mathrm{N3} \\
 & \mathrm{N5} \leftarrow \mathrm{N1111} \\
\mathrm{L1000} & \mathrm{IF\ N2222} \neq 0\ \mathrm{GOTO\ L2000} \\
 & \mathrm{GOTO\ L3000} \\
\mathrm{L2000} & \mathrm{N2222} \leftarrow \mathrm{N2222} \dot{-} 1 \\
 & \mathrm{N5} \leftarrow \mathrm{N5} + 1 \\
 & \mathrm{GOTO\ L1000} \\
\mathrm{L3000} & \mathrm{SKIP}
\end{array}$$

donde las variables N1111, N2222 y los labels L1000, L2000, L3000 solo seran usados aqui, es decir no apareceran en el resto de nuestro programa $\mathcal{P}$. Notese que este programa cuando es corrido termina dejando en la variable N5 la suma de los contenidos de las variables N16 y N3 y modifica el contenido de las variables N1111 y N2222, lo cual no traera problemas ya que N1111 y N2222 no se usan en el resto de $\mathcal{P}$. La variables N1111 y N2222 son auxiliares y se usan justamente para preservar el valor de las variables N16 y N3 ya que ellas son variables protagonistas de nuestro programa $\mathcal{P}$ y en esta instancia no queremos alterar su contenido sino solo realizar la asignacion $\mathrm{N5} \leftarrow \mathrm{N16} + \mathrm{N3}$. Dejamos al lector explicar por que es necesario para que la simulacion sea correcta que los labels L1000, L2000 y L3000 no sean usados en el resto de $\mathcal{P}$.

Es decir el programa anterior simula la instruccion $\mathrm{N5} \leftarrow \mathrm{N16} + \mathrm{N3}$ que no podiamos usar por no ser una instruccion de $\mathcal{S}^{\Sigma}$, con un costo bastante bajo, es decir el costo de convenir en no usar en el resto de $\mathcal{P}$ las variables N1111 y N2222 ni los labels L1000, L2000 y L3000.

Ahora supongamos que seguimos escribiendo el programa $\mathcal{P}$ y nos hace falta simular la instruccion $\mathrm{N20} \leftarrow \mathrm{N1} + \mathrm{N14}$. Entonces es claro que podriamos modificar el programa que simulaba $\mathrm{N5} \leftarrow \mathrm{N16} + \mathrm{N3}$ haciendole reemplazos adecuados a sus variables y labels. Por ejemplo podriamos escribir

$$\begin{array}{ll}
 & \mathrm{N9999} \leftarrow \mathrm{N1} \\
 & \mathrm{N8888} \leftarrow \mathrm{N14} \\
 & \mathrm{N20} \leftarrow \mathrm{N9999} \\
\mathrm{L1001} & \mathrm{IF\ N8888} \neq 0 \mathrm{\ GOTO\ L2002} \\
 & \mathrm{GOTO\ L3003} \\
\mathrm{L2002} & \mathrm{N8888} \leftarrow \mathrm{N8888} \dot{-} 1 \\
 & \mathrm{N20} \leftarrow \mathrm{N20} + 1 \\
 & \mathrm{GOTO\ L1001} \\
\mathrm{L3003} & \mathrm{SKIP}
\end{array}$$

donde N9999, N8888, L1001, L2002 y L3003 solo seran usados aqui, es decir no apareceran en el resto de nuestro programa $\mathcal{P}$.

Consideremos el siguiente "molde" que llamaremos M

$$\begin{array}{ll}
 & \mathrm{V4} \leftarrow \mathrm{V2} \\
 & \mathrm{V5} \leftarrow \mathrm{V3} \\
 & \mathrm{V1} \leftarrow \mathrm{V4} \\
\mathrm{A1} & \mathrm{IF\ V5} \neq 0 \mathrm{\ GOTO\ A2} \\
 & \mathrm{GOTO\ A3} \\
\mathrm{A2} & \mathrm{V5} \leftarrow \mathrm{V5} \dot{-} 1 \\
 & \mathrm{V1} \leftarrow \mathrm{V1} + 1 \\
 & \mathrm{GOTO\ A1} \\
\mathrm{A3} & \mathrm{SKIP}
\end{array}$$

Como puede notarse, cuando reemplazamos en M

- cada ocurrencia de V1 por N5
- cada ocurrencia de V2 por N16
- cada ocurrencia de V3 por N3
- cada ocurrencia de V4 por N1111
- cada ocurrencia de V5 por N2222
- cada ocurrencia de A1 por L1000
- cada ocurrencia de A2 por L2000
- cada ocurrencia de A3 por L3000

obtenemos el programa que simulaba la instruccion $\mathrm{N5} \leftarrow \mathrm{N16} + \mathrm{N3}$ dentro de $\mathcal{P}$. Similarmente, cuando reemplazamos en M

- cada ocurrencia de V1 por N20
- cada ocurrencia de V2 por N1

- cada ocurrencia de V3 por N14

- cada ocurrencia de V4 por N9999

- cada ocurrencia de V5 por N8888

- cada ocurrencia de A1 por L1001

- cada ocurrencia de A2 por L2002

- cada ocurrencia de A3 por L3003

obtenemos el programa que simulaba la instruccion N20 $\leftarrow$ N1 + N14 dentro de $\mathcal{P}$. La practicidad de tener el molde M cae de maduro. Ahora en caso de necesitar una instruccion del tipo N$\bar{k}$ $\leftarrow$ N$\bar{n}$ + N$\bar{m}$ solo tenemos que reemplazar en M

- cada ocurrencia de V1 por N$\bar{k}$

- cada ocurrencia de V2 por N$\bar{n}$

- cada ocurrencia de V3 por N$\bar{m}$

y reemplazar la variable auxiliar de M y los labels auxiliares de M por una variable concreta y tres labels concretos que no se usen en el programa que estamos realizando. El programa asi obtenido simulara a la instruccion N$\bar{k}$ $\leftarrow$ N$\bar{n}$ + N$\bar{m}$.

En la gerga computacional el molde M suele llamarse *macro* y los programas obtenidos luego de realizar los reemplazos son llamados *expansiones de* M. Notese que $Ti(M)$ = PALABRA ya que, como en el caso de los programas, podemos pensar que escribimos a M linea por linea para facilitar su manejo pero que en realidad es una sola palabra, a saber:

V1$\leftarrow$V2V4$\leftarrow$V3A1IFV4$\neq$0GOTOA2GOTOA3A2V4$\leftarrow$V4$\dot{-}$1V1$\leftarrow$V1+1GOTOA1A3SKIP

Es decir, como objeto matematico, M es una palabra. A las palabras de la forma V$\bar{n}$, con $n \in \mathbf{N}$, las llamaremos *variables numericas de macro.* A las palabras de la forma W$\bar{n}$, con $n \in \mathbf{N}$, las llamaremos *variables alfabeticas de macro* y a las palabras de la forma A$\bar{n}$, con $n \in \mathbf{N}$, las llamaremos *labels de macro.* Nuestro macro M no tiene variables alfabeticas de macro pero otros macros por supuesto pueden tener este tipo de variables.

Las variables V1, V2 y V3 son llamadas *variables oficiales* de M y V4 y V5 son llamadas *variables auxiliares* de M. Tambien A1, A2 y A3 son llamados *labels auxiliares* de M ya que son usados solo para su funcionamiento interno y no tienen vinculacion con los labels del programa en el cual se realizara la expansion de M.

En el siguiente ejemplo veremos un macro que tiene un label que no es auxiliar sino oficial. Sea $\Sigma = \{@, !\}$. Supongamos que estamos escribiendo un programa $\mathcal{P}'$ y nos hace falta simular instrucciones de la forma

$$\text{IF } |\text{P}\bar{n}| \leq \text{N}\bar{m} \text{ GOTO L}\bar{k}$$

(por supuesto estas instrucciones no pertenecen al lenguaje $\mathcal{S}^{\Sigma}$ pero deberia quedar claro como funcionan). Entonces podemos tomar el macro M':

	W2 $\leftarrow$ W1
	V2 $\leftarrow$ V1
A4	IF W2 BEGINS @ GOTO A2
	IF W2 BEGINS ! GOTO A2
	GOTO A1
A2	IF V2 $\neq$ 0 GOTO A3
	GOTO A5
A3	W2 $\leftarrow^{\curvearrowright}$ W2
	V2 $\leftarrow$ V2$\dot{-}$1
	GOTO A4
A5	SKIP

el cual tiene

- variables oficiales W1 y V1 (correspondientes a P$\bar{n}$ y N$\bar{m}$)

- variable auxiliares W2 y V2

- labels auxiliares A2, A3, A4 y A5

- un label oficial A1 (correspondiente a L$\bar{k}$)

Una descripcion intuitiva del macro M' seria

$$\text{IF } |\text{W1}| \leq \text{V1 GOTO A1}$$

Notese que en las primeras dos lineas el macro M' guarda los valores de las variables oficiales W1 y V1 en las variables auxiliares W2 y V2, y sigue trabajando con las auxiliares. Esto es para preservar el valor de las variables oficiales. Dado que $\Sigma = \{@, !\}$, las dos siguientes lineas sirven para decidir si el contenido de W2 es ε o no. Dejamos al lector entender el resto del funcionamiento de M'.

Para dar un ejemplo de como usariamos a M', supongamos que para seguir escribiendo nuestro programa $\mathcal{P}'$ nos hace falta simular la instruccion

$$\text{IF } |\text{P5}| \leq \text{N14 GOTO L1}$$

y supongamos que las variables P1000 y N1000 y los labels L6666, L7777, L8888 y L9999 no se usaron hasta el momento en $\mathcal{P}'$. Entonces podemos reemplazar en M'

- cada ocurrencia de W1 por P5

- cada ocurrencia de V1 por N14

- cada ocurrencia de W2 por P1000

- cada ocurrencia de V2 por N1000

- cada ocurrencia de A1 por L1
- cada ocurrencia de A2 por L6666
- cada ocurrencia de A3 por L7777
- cada ocurrencia de A4 por L8888
- cada ocurrencia de A5 por L9999

y la expansion de M' asi obtenida simulara la instruccion IF $|\text{P5}| \leq$ N14 GOTO L1. Cabe destacar que para asegurarnos que la simulacion funcione, tambien deberemos no usar en el resto de $\mathcal{P}'$ las variables P1000 y N1000 y los labels L6666, L7777, L8888 y L9999.

Es decir M' funciona como un molde con el cual haciendo reemplazos adecuados podemos simular cualquier instruccion del tipo IF $|\text{P}\bar{n}| \leq \text{N}\bar{m}$ GOTO L$\bar{k}$, con $n, m, k \in \mathbf{N}$.

Deberia quedar claro el caracter oficial del label A1 en M' ya que el label por el que se lo reemplaza para hacer la expansion es uno de los labels protagonistas del programa que se esta escribiendo.

Cabe destacar que las expansiones de M' no son programas ya que si bien son concatenaciones de instrucciones, no cumplen la ley de los GOTO (llamada (G) en la definicion de programa) respecto del label que reemplazo a A1.

Nota: Siempre supondremos que la primera instruccion de los macros no es labelada. Esto es porque muchas veces cuando expandamos un macro nos interesara labelar la primera instruccion de dicha expansion. Por supuesto, esto es facil de conseguir ya que si M es un macro, entonces SKIPM es tambien un macro que posee las mismas propiedades.

Como hemos visto recien hay dos tipos de macros:

- los de asignacion que cuando son expandidos nos dan un programa que simula la asignacion a una variable dada del resultado de aplicar una funcion a los contenidos de ciertas otras variables; y
- los de tipo IF que cuando son expandidos nos dan un programa salvo por la ley (G), el cual direcciona al label que fue a reemplazar a A1 cuando se cumple cierta propiedad (predicado) relativa a los contenidos de las variables que fueron a reemplazar a las variables oficiales.

Ejemplo concreto de uso de macros Ya vimos recien que la palabra

$$\begin{array}{ll} & \mathrm{V4} \leftarrow \mathrm{V2} \\ & \mathrm{V5} \leftarrow \mathrm{V3} \\ & \mathrm{V1} \leftarrow \mathrm{V4} \\ \mathrm{A1} & \mathrm{IF\ V5} \neq 0 \mathrm{\ GOTO\ A2} \\ & \mathrm{GOTO\ A3} \\ \mathrm{A2} & \mathrm{V5} \leftarrow \mathrm{V5} \dot{-} 1 \\ & \mathrm{V1} \leftarrow \mathrm{V1} + 1 \\ & \mathrm{GOTO\ A1} \\ \mathrm{A3} & \mathrm{SKIP} \end{array}$$

es un macro que sirve para simular instrucciones de la forma $\mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{n} + \mathrm{N}\bar{m}$. Notemos que este macro es de asignacion ya que cuando es expandido nos da un programa que simula la asignacion a una variable dada del resultado de aplicar una funcion a los contenidos de ciertas otras variables. En este caso la funcion es $SUMA = \lambda xy[x+y]$ por lo cual usaremos $[\mathrm{V3} \leftarrow SUMA(\mathrm{V1}, \mathrm{V2})]$ para denotar a dicho macro. Usaremos este macro para dar un programa $\mathcal{P}$ que compute a la funcion $\lambda xy[x.y]$. Notese que podemos tomar $\mathcal{P}$ igual al siguiente programa

$$\begin{array}{ll} \mathrm{L1} & \mathrm{IF\ N2} \neq 0 \mathrm{\ GOTO\ L2} \\ & \mathrm{GOTO\ L3} \\ \mathrm{L2} & [\mathrm{N3} \leftarrow SUMA(\mathrm{N3}, \mathrm{N1})] \\ & \mathrm{N2} \leftarrow \mathrm{N2} \dot{-} 1 \\ & \mathrm{GOTO\ L1} \\ \mathrm{L3} & \mathrm{N1} \leftarrow \mathrm{N3} \end{array}$$

donde $[\mathrm{N3} \leftarrow SUMA(\mathrm{N3}, \mathrm{N1})]$ es una expansion del macro $[\mathrm{V3} \leftarrow SUMA(\mathrm{V1}, \mathrm{V2})]$ hecha haciendo el reemplazo de las variables oficiales $\mathrm{V3}, \mathrm{V1}$ y $\mathrm{V2}$ por $\mathrm{N3}, \mathrm{N3}$ y N1, respectivamente, y haciendo reemplazos adecuados de sus variables y labels auxiliares. Hay muchas formas de hacer los reemplazos de variables y labels auxiliares pero en general no lo especificaremos explicitamente cuando expandamos un macro ya que es facil imaginar como hacerlo dependiendo del programa que estemos realizando. Por ejemplo en el caso de $\mathcal{P}$ podriamos hacer los siguientes reemplazos:

- cada ocurrencia de V4 por N1111
- cada ocurrencia de V5 por N2222
- cada ocurrencia de A1 por L1000
- cada ocurrencia de A2 por L2000
- cada ocurrencia de A3 por L3000

y claramente esto no afectara la "logica" o "idea" de nuestro programa $\mathcal{P}$. De esta forma la expansion $[\mathrm{N3} \leftarrow SUMA(\mathrm{N3}, \mathrm{N1})]$ es el siguiente programa:

$$\begin{array}{ll} & \mathrm{N1111} \leftarrow \mathrm{N1} \\ & \mathrm{N2222} \leftarrow \mathrm{N3} \\ & \mathrm{N3} \leftarrow \mathrm{N1111} \\ \mathrm{L1000} & \mathrm{IF\ N2222} \neq 0 \mathrm{\ GOTO\ L2000} \\ & \mathrm{GOTO\ L3000} \\ \mathrm{L2000} & \mathrm{N2222} \leftarrow \mathrm{N2222} \dot{-} 1 \\ & \mathrm{N3} \leftarrow \mathrm{N3} + 1 \\ & \mathrm{GOTO\ L1000} \\ \mathrm{L3000} & \mathrm{SKIP} \end{array}$$

el cual por supuesto esta escrito con espacios y en forma vertical pero es una mera palabra. Tenemos entonces que $\mathcal{P}$ es el programa:

$$\begin{array}{ll} \mathrm{L1} & \mathrm{IF\ N2} \neq 0 \mathrm{\ GOTO\ L2} \\ & \mathrm{GOTO\ L3} \\ \mathrm{L2} & \mathrm{N1111} \leftarrow \mathrm{N1} \\ & \mathrm{N2222} \leftarrow \mathrm{N3} \\ & \mathrm{N3} \leftarrow \mathrm{N1111} \\ \mathrm{L1000} & \mathrm{IF\ N2222} \neq 0 \mathrm{\ GOTO\ L2000} \\ & \mathrm{GOTO\ L3000} \\ \mathrm{L2000} & \mathrm{N2222} \leftarrow \mathrm{N2222} \dot{-} 1 \\ & \mathrm{N3} \leftarrow \mathrm{N3} + 1 \\ & \mathrm{GOTO\ L1000} \\ \mathrm{L3000} & \mathrm{SKIP} \\ & \mathrm{N2} \leftarrow \mathrm{N2} \dot{-} 1 \\ & \mathrm{GOTO\ L1} \\ \mathrm{L3} & \mathrm{N1} \leftarrow \mathrm{N3} \end{array}$$

el cual por supuesto esta escrito con espacios y en forma vertical pero es una mera palabra.

Macros asociados a funciones Σ-computables Dada una funcion $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to \omega$, usaremos

$$\left[\mathrm{V}\overline{n+1} \leftarrow f(\mathrm{V}1, ..., \mathrm{V}\bar{n}, \mathrm{W}1, ..., \mathrm{W}\bar{m})\right]$$

para denotar un macro M el cual cumpla las siguientes propiedades. Cabe destacar que no siempre existira dicho macro, es decir solo para ciertas funciones $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to \omega$ habra un tal macro.

(1) Las variables oficiales de M son $\mathrm{V}1, ..., \mathrm{V}\bar{n}, \mathrm{V}\overline{n+1}, \mathrm{W}1, ..., \mathrm{W}\bar{m}$

(2) M no tiene labels oficiales

(3) Si reemplazamos:

(a) las variables oficiales de M (i.e. $\text{V}1,...,\text{V}\bar{n},\text{V}\overline{n+1},\text{W}1,...,\text{W}\bar{m}$) por variables concretas

$$\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{N}\overline{k_{n+1}},\text{P}\overline{j_1},...,\text{P}\overline{j_m}$$

(elejidas libremente, es decir los numeros $k_1,...,k_{n+1},j_1,...,j_m$ son cualesquiera)

(b) las variables auxiliares de M por variables concretas (distintas de a dos) y NO pertenecientes a la lista $\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{N}\overline{k_{n+1}},\text{P}\overline{j_1},...,\text{P}\overline{j_m}$

(c) los labels auxiliares de M por labels concretos (distintos de a dos)

Entonces la palabra asi obtenida es un programa de $\mathcal{S}^\Sigma$ que denotaremos con

$$\left[\text{N}\overline{k_{n+1}} \leftarrow f(\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m})\right]$$

el cual debe tener la siguiente propiedad:

- Si hacemos correr $\left[\text{N}\overline{k_{n+1}} \leftarrow f(\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m})\right]$ partiendo de un estado e que le asigne a las variables $\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m}$ valores $x_1,...,x_n,\alpha_1,...,\alpha_m$, entonces independientemente de los valores que les asigne e al resto de las variables (incluidas las que fueron a reemplazar a las variables auxiliares de M) se dara que

  i. si $(x_1,...,x_n,\alpha_1,...,\alpha_m) \notin D_f$, entonces $\left[\text{N}\overline{k_{n+1}} \leftarrow f(\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m})\right]$ no se detiene

  ii. si $(x_1,...,x_n,\alpha_1,...,\alpha_m) \in D_f$, entonces $\left[\text{N}\overline{k_{n+1}} \leftarrow f(\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m})\right]$ se detiene (i.e. intenta realizar la siguiente a su ultima instrucion) y llega a un estado e' el cual cumple:

     A. e' le asigna a $\text{N}\overline{k_{n+1}}$ el valor $f(x_1,...,x_n,\alpha_1,...,\alpha_m)$

     B. e' solo puede diferir de e en los valores que le asigna a $\text{N}\overline{k_{n+1}}$ o a las variables que fueron a reemplazar a las variables auxiliares de M. Al resto de las variables, incluidas $\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m}$ no las modifica (salvo en el caso de que alguna $\text{N}\overline{k_i}$ sea la variable $\text{N}\overline{k_{n+1}}$, situacion en la cual el valor final de la variable $\text{N}\overline{k_i}$ sera $f(x_1,...,x_n,\alpha_1,...,\alpha_m)$)

El programa $\left[\text{N}\overline{k_{n+1}} \leftarrow f(\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m})\right]$ es comunmente llamado la expansion del macro $\left[\text{V}\overline{n+1} \leftarrow f(\text{V}1,...,\text{V}\bar{n},\text{W}1,...,\text{W}\bar{m})\right]$ con respecto a la eleccion de variables y labels realizada.

Tambien, dada una funcion $f : D_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow \Sigma^*$, con

$$\left[\text{W}\overline{m+1} \leftarrow f(\text{V}1,...,\text{V}\bar{n},\text{W}1,...,\text{W}\bar{m})\right]$$

denotaremos un macro el cual cumpla condiciones analogas a las descriptas recien. Dejamos al lector escribirlas en detalle para este caso.

Proposition 87 *(a) Sea $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to \omega$ una funcion Σ-computable. Entonces en $\mathcal{S}^\Sigma$ hay un macro*

$$\left[\mathrm{V}\overline{n+1} \leftarrow f(\mathrm{V}1, ..., \mathrm{V}\bar{n}, \mathrm{W}1, ..., \mathrm{W}\bar{m})\right]$$

(b) Sea $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to \Sigma^*$ una funcion Σ-computable. Entonces en $\mathcal{S}^\Sigma$ hay un macro

$$\left[\mathrm{W}\overline{m+1} \leftarrow f(\mathrm{V}1, ..., \mathrm{V}\bar{n}, \mathrm{W}1, ..., \mathrm{W}\bar{m})\right]$$

Proof. Probaremos (b) La prueba de (a) es similar. Sea $\mathcal{P}$ un programa que compute a f. Tomemos un k tal que $k \geq n, m$ y tal que todas las variables y labels de $\mathcal{P}$ estan en el conjunto

$$\{\mathrm{N}1, ..., \mathrm{N}\bar{k}, \mathrm{P}1, ..., \mathrm{P}\bar{k}, \mathrm{L}1, ..., \mathrm{L}\bar{k}\}.$$

Sea $\mathcal{P}'$ la palabra que resulta de reemplazar en $\mathcal{P}$:

- la variable $\mathrm{N}\bar{j}$ por $\mathrm{V}\overline{n+j}$, para cada $j = 1, ..., k$
- la variable $\mathrm{P}\bar{j}$ por $\mathrm{W}\overline{m+j}$, para cada $j = 1, ..., k$
- el label $\mathrm{L}\bar{j}$ por $\mathrm{A}\bar{j}$, para cada $j = 1, ..., k$

Notese que

$$\begin{array}{l}
\mathrm{V}\overline{n+1} \leftarrow \mathrm{V}1 \\
\quad\vdots \\
\mathrm{V}\overline{n+n} \leftarrow \mathrm{V}\bar{n} \\
\mathrm{V}\overline{n+n+1} \leftarrow 0 \\
\quad\vdots \\
\mathrm{V}\overline{n+k} \leftarrow 0 \\
\mathrm{W}\overline{m+1} \leftarrow \mathrm{W}1 \\
\quad\vdots \\
\mathrm{W}\overline{m+m} \leftarrow \mathrm{W}\bar{m} \\
\mathrm{W}\overline{m+m+1} \leftarrow \varepsilon \\
\quad\vdots \\
\mathrm{W}\overline{m+k} \leftarrow \varepsilon \\
\mathcal{P}'
\end{array}$$

es el macro buscado, el cual tendra sus variables auxiliares y labels en la lista

$$\mathrm{V}\overline{n+1}, ..., \mathrm{V}\overline{n+k}, \mathrm{W}\overline{m+2}, ..., \mathrm{W}\overline{m+k}, \mathrm{A}1, ..., \mathrm{A}\bar{k}.$$

■

Dejamos al lector probar la resiproca de la proposicion anterior, es decir que si $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to \omega$ es tal que en $\mathcal{S}^\Sigma$ hay un macro

$$\left[\mathrm{V}\overline{n+1} \leftarrow f(\mathrm{V}1, ..., \mathrm{V}\bar{n}, \mathrm{W}1, ..., \mathrm{W}\bar{m})\right]$$

entonces f es Σ-computable

Macros asociados a predicados Σ-computables Dado un predicado $P : D_P \subseteq \omega^n \times \Sigma^{*m} \rightarrow \omega$, usaremos

$$[\text{IF } P(\text{V}1,...,\text{V}\bar{n},\text{W}1,...,\text{W}\bar{m}) \text{ GOTO A1}]$$

para denotar un macro M el cual cumpla las siguientes propiedades. Cabe destacar que no siempre existira dicho macro, es decir solo para ciertos predicados $P : D_P \subseteq \omega^n \times \Sigma^{*m} \rightarrow \omega$ habra un tal macro.

(1) Las variables oficiales de M son $\text{V}1,...,\text{V}\bar{n},\text{W}1,...,\text{W}\bar{m}$

(2) A1 es el unico label oficial de M

(3) Si reemplazamos:

   (a) las variables oficiales de M (i.e. $\text{V}1,...,\text{V}\bar{n},\text{W}1,...,\text{W}\bar{m}$) por variables concretas
   $$\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m}$$
   (elejidas libremente, es decir los numeros $k_1,...,k_n,j_1,...,j_m$ son cualesquiera)

   (b) el label oficial A1 por el label concreto $\text{L}\bar{k}$ (elejido libremente, es decir k es cualquier elemento de $\mathbf{N}$)

   (c) las variables auxiliares de M por variables concretas (distintas de a dos) y NO pertenecientes a la lista $\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m}$

   (d) los labels auxiliares de M por labels concretos (distintos de a dos) y ninguno igual a $\text{L}\bar{k}$

   Entonces la palabra asi obtenida es un programa de $\mathcal{S}^\Sigma$ (salvo por la ley de los GOTO respecto de $\text{L}\bar{k}$) que denotaremos con
   $$\left[\text{IF } P(\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m}) \text{ GOTO L}\bar{k}\right]$$
   el cual debe tener la siguiente propiedad:

   - Si hacemos correr $\left[\text{IF } P(\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m}) \text{ GOTO L}\bar{k}\right]$ partiendo de un estado e que le asigne a las variables $\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m}$ valores $x_1,...,x_n,\alpha_1,...,\alpha_m$, entonces independientemente de los valores que les asigne e al resto de las variables (incluidas las que fueron a reemplazar a las variables auxiliares de M) se dara que

      i. si $(x_1,...,x_n,\alpha_1,...,\alpha_m) \notin D_P$, entonces $\left[\text{IF } P(\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m}) \text{ GOTO L}\bar{k}\right]$ no se detiene

      ii. si $(x_1,...,x_n,\alpha_1,...,\alpha_m) \in D_P$ y $P(x_1,...,x_n,\alpha_1,...,\alpha_m) = 1$, entonces luego de una cantidad finita de pasos, $\left[\text{IF } P(\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m}) \text{ GOTO L}\bar{k}\right]$ direcciona al label $\text{L}\bar{k}$ quedando en un estado e' el cual solo puede diferir de e en los valores que le asigna a las variables que fueron a reemplazar a las variables auxiliares de M. Al resto de las variables, incluidas $\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m}$ no las modifica

iii. si $(x_1,...,x_n,\alpha_1,...,\alpha_m) \in D_P$ y $P(x_1,...,x_n,\alpha_1,...,\alpha_m) = 0$, entonces luego de una cantidad finita de pasos, $\left[\text{IF } P(\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m}) \text{ GOTO L}\bar{k}\right]$ se detiene (i.e. intenta realizar la siguiente a su ultima instruccion) quedando en un estado e' el cual solo puede diferir de e en los valores que le asigna a las variables que fueron a reemplazar a las variables auxiliares de M. Al resto de las variables, incluidas $\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m}$ no las modifica

La palabra $\left[\text{IF } P(\text{N}\overline{k_1},...,\text{N}\overline{k_n},\text{P}\overline{j_1},...,\text{P}\overline{j_m}) \text{ GOTO L}\bar{k}\right]$ es llamada la expansion del macro con respecto a la eleccion de variables y labels realizada

Proposition 88 *Sea $P : D_P \subseteq \omega^n \times \Sigma^{*m} \to \omega$ un predicado Σ-computable. Entonces en $\mathcal{S}^\Sigma$ hay un macro*

$$[\text{IF } P(\text{V1},...,\text{V}\bar{n},\text{W1},...,\text{W}\bar{m}) \text{ GOTO A1}]$$

Proof. Por (a) de la proposicion anterior tenemos un macro $\left[\text{V}\overline{n+1} \leftarrow P(\text{V1},...,\text{V}\bar{n},\text{W1},...,\text{W}\bar{m})\right]$. Notese que la palabra

$$\left[\text{V}\overline{n+1} \leftarrow P(\text{V1},...,\text{V}\bar{n},\text{W1},...,\text{W}\bar{m})\right] \text{IFV}\overline{n+1}{\neq}0\text{GOTOA1}$$

es el macro buscado. ■

Dejamos al lector probar la resiproca de la proposicion anterior, es decir si $P : D_P \subseteq \omega^n \times \Sigma^{*m} \to \omega$ es tal que en $\mathcal{S}^\Sigma$ hay un macro

$$[\text{IF } P(\text{V1},...,\text{V}\bar{n},\text{W1},...,\text{W}\bar{m}) \text{ GOTO A1}]$$

entonces P es Σ-computable.

3.3.5 Conjuntos Σ-enumerables

Ya que la nocion de funcion Σ-computable es el modelo matematico Neumanniano o imperativo del concepto de funcion Σ-efectivamente computable, nos podriamos preguntar entonces cual es el modelo matematico Neumanniano del concepto de conjunto Σ-efectivamente enumerable. Si prestamos atencion a la definicion de conjunto Σ-efectivamente enumerable, notaremos que depende de la existencia de ciertas funciones Σ-efectivamente computables por lo cual la siguiente definicion cae de maduro:

Un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ sera llamado Σ-*enumerable* cuando sea vacio o haya una funcion $F : \omega \to \omega^n \times \Sigma^{*m}$ tal que $I_F = S$ y $F_{(i)}$ sea Σ-computable, para cada $i \in \{1,...,n+m\}$.

Deberia entonces quedar claro que si el concepto de funcion Σ-computable modeliza correctamente al concepto de funcion Σ-efectivamente computable, entonces el concepto de conjunto Σ-enumerable recien definido modeliza correctamente al concepto de conjunto Σ-efectivamente enumerable. Notese que segun la definicion que acabamos de escribir, un conjunto no vacio $S \subseteq \omega^n \times \Sigma^{*m}$ es Σ-enumerable si y solo si hay programas $\mathcal{P}_1,...,\mathcal{P}_{n+m}$ tales que

- $\text{Dom}(\Psi^{1,0,\#}_{\mathcal{P}_1}) = ... = \text{Dom}(\Psi^{1,0,\#}_{\mathcal{P}_n}) = \omega$

- $\text{Dom}(\Psi^{1,0,*}_{\mathcal{P}_{n+1}}) = ... = \text{Dom}(\Psi^{1,0,*}_{\mathcal{P}_n+m}) = \omega$

- $S = \text{Im}[\Psi^{1,0,\#}_{\mathcal{P}_1}, ..., \Psi^{1,0,\#}_{\mathcal{P}_n}, \Psi^{1,0,*}_{\mathcal{P}_{n+1}}, ..., \Psi^{1,0,*}_{\mathcal{P}_n+m}]$

Como puede notarse los programas $\mathcal{P}_1, ..., \mathcal{P}_{n+m}$ puestos en paralelo a funcionar desde el estado $\|x\|$ producen en forma natural un procedimiento efectivo (con dato de entrada $x \in \omega$) que enumera a S. Por supuesto podemos decir que en tal caso los programas $\mathcal{P}_1, ..., \mathcal{P}_{n+m}$ enumeran a S. La siguiente proposicion muestra que tambien las cosas se pueden hacer con un solo programa

Proposition 89 *Sea $S \subseteq \omega^n \times \Sigma^{*m}$ un conjunto no vacio. Entonces son equivalentes:*

(1) S es Σ-enumerable

(2) Hay un programa $\mathcal{P} \in \text{Pro}^\Sigma$ tal que:

(a) Para cada $x \in \omega$, tenemos que $\mathcal{P}$ se detiene partiendo desde el estado $\|x\|$ y llega a un estado $\|x_1, ...x_n, \alpha_1, ..., \alpha_m\|$, donde $(\vec{x}, \vec{\alpha}) \in S$.

(b) Para cada $(\vec{x}, \vec{\alpha}) \in S$ hay un $x \in \omega$ tal que $\mathcal{P}$ se detiene partiendo desde el estado $\|x\|$ y llega al estado $\|x_1, ...x_n, \alpha_1, ..., \alpha_m\|$

(3) Hay un programa $\mathcal{P} \in \text{Pro}^\Sigma$ tal que:

(a) Para cada $x \in \omega$, tenemos que $\mathcal{P}$ se detiene partiendo desde el estado $\|x\|$ y llega a un estado de la forma $((x_1, ..., x_n, y_1, ...), (\alpha_1, ..., \alpha_m, \beta_1, ...))$, donde $(x_1, ..., x_n, \alpha_1, ..., \alpha_m) \in S$.

(b) Para cada $(x_1, ...x_n, \alpha_1, ..., \alpha_m) \in S$ hay un $x \in \omega$ tal que $\mathcal{P}$ se detiene partiendo desde el estado $\|x\|$ y llega a un estado de la forma $((x_1, ..., x_n, y_1, ...), (\alpha_1, ..., \alpha_m, \beta_1, ...))$

Proof. (1)$\Rightarrow$(2). Ya que S es no vacio, por definicion tenemos que hay una $F : \omega \to \omega^n \times \Sigma^{*m}$ tal que $I_F = S$ y $F_{(i)}$ es Σ-computable, para cada $i \in \{1, ..., n+m\}$. Por la Proposicion 87 tenemos que existen macros:

$$\begin{gathered}
[\text{V2} \leftarrow F_{(1)}(\text{V1})] \\
\vdots \\
[\text{V2} \leftarrow F_{(n)}(\text{V1})] \\
[\text{W1} \leftarrow F_{(n+1)}(\text{V1})] \\
\vdots \\
[\text{W1} \leftarrow F_{(n+m)}(\text{V1})]
\end{gathered}$$

Sea $\mathcal{Q}$ el siguiente programa:

$$\begin{array}{c}
\left[\mathrm{P}\overline{m} \leftarrow F_{(n+m)}(\mathrm{N}1)\right] \\
\vdots \\
\left[\mathrm{P}1 \leftarrow F_{(n+1)}(\mathrm{N}1)\right] \\
\left[\mathrm{N}\overline{n} \leftarrow F_{(n)}(\mathrm{N}1)\right] \\
\vdots \\
\left[\mathrm{N}1 \leftarrow F_{(1)}(\mathrm{N}1)\right]
\end{array}$$

donde se supone que las expansiones de los macros usados son hechas usando variables auxiliares no pertenecientes a la lista $\mathrm{N}1, ..., \mathrm{N}\overline{n}, \mathrm{P}1, ..., \mathrm{P}\overline{m}$ (por supuesto, dada la fortaleza de nuestros macros se puede usa una misma variable auxiliar para dos distintas expansiones), y tambien se supone que los labels auxiliares usados en dichas expansiones son todos distintos, es decir no usamos el mismo label auxiliar en dos expansiones distintas (por que?).

Sea k tal que las variables de $\mathcal{Q}$ estan todas en la lista $\mathrm{N}1, ..., \mathrm{N}\bar{k}, \mathrm{P}1, ..., \mathrm{P}\bar{k}$. Sea $\mathcal{P}$ el siguiente programa:

$$\mathcal{Q}\mathrm{N}\overline{n+1} \leftarrow 0\mathrm{N}\overline{n+2} \leftarrow 0...\mathrm{N}\bar{k} \leftarrow 0\mathrm{P}\overline{m+1} \leftarrow \varepsilon\mathrm{P}\overline{m+2} \leftarrow \varepsilon...\mathrm{P}\bar{k} \leftarrow \varepsilon$$

Dejamos al lector corroborar que el programa $\mathcal{P}$ cumple las propiedades a y b

(2)⇒(3). Directo.

(3)⇒(1). Supongamos $\mathcal{P} \in \mathrm{Pro}^{\Sigma}$ cumple a y b de (3). Sean

$$\begin{array}{rcl}
\mathcal{P}_1 & = & \mathcal{P}\mathrm{N}1 \leftarrow \mathrm{N}1 \\
\mathcal{P}_2 & = & \mathcal{P}\mathrm{N}1 \leftarrow \mathrm{N}2 \\
& \vdots & \\
\mathcal{P}_n & = & \mathcal{P}\mathrm{N}1 \leftarrow \mathrm{N}\overline{n} \\
\mathcal{P}_{n+1} & = & \mathcal{P}\mathrm{P}1 \leftarrow \mathrm{P}1 \\
\mathcal{P}_{n+2} & = & \mathcal{P}\mathrm{P}1 \leftarrow \mathrm{P}2 \\
& \vdots & \\
\mathcal{P}_{n+m} & = & \mathcal{P}\mathrm{P}1 \leftarrow \mathrm{P}\overline{m}
\end{array}$$

Definamos

$$\begin{array}{rcl} F_1 & = & \Psi_{\mathcal{P}_1}^{1,0,\#} \\ F_2 & = & \Psi_{\mathcal{P}_2}^{1,0,\#} \\ & \vdots & \\ F_n & = & \Psi_{\mathcal{P}_n}^{1,0,\#} \\ F_{n+1} & = & \Psi_{\mathcal{P}_{n+1}}^{1,0,*} \\ F_{n+2} & = & \Psi_{\mathcal{P}_{n+2}}^{1,0,*} \\ & \vdots & \\ F_{n+m} & = & \Psi_{\mathcal{P}_{n+m}}^{1,0,*} \end{array}$$

Notese que cada F_i es Σ-computable y tiene dominio igual a ω. Sea $F = [F_1, ..., F_{n+m}]$. Tenemos por definicion que $D_F = \omega$ y ya que $F_{(i)} = F_i$, para cada $i = 1, ..., n+m$ tenemos que cada $F_{(i)}$ es Σ-computable. Dejamos al lector verificar que $I_F = S$ ■

Cuando un programa $\mathcal{P}$ cumpla las propiedades dadas en (3) de la proposicion anterior respecto de un conjunto S, diremos que $\mathcal{P}$ *enumera* a S.

Cabe destacar que (3)$\Rightarrow$(1) de la proposicion anterior es muy util a la hora de probar que un conjunto dado es Σ-enumerable.

3.3.6 Conjuntos Σ-computables

La version imperativa del concepto de conjunto Σ-efectivamente computable es facil de dar: un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ sera llamado Σ-*computable* cuando la funcion $\chi_S^{\omega^n \times \Sigma^{*m}}$ sea Σ-computable. O sea que $S \subseteq \omega^n \times \Sigma^{*m}$ es Σ-computable sii hay un programa $\mathcal{P} \in \mathrm{Pro}^\Sigma$ el cual computa a $\chi_S^{\omega^n \times \Sigma^{*m}}$, es decir:

- Si $(\vec{x}, \vec{\alpha}) \in S$, entonces $\mathcal{P}$ se detiene partiendo desde $\|x_1, ...x_n, \alpha_1, ..., \alpha_m\|$ y la variable N1 queda con contenido igual a 1
- Si $(\vec{x}, \vec{\alpha}) \in (\omega^n \times \Sigma^{*m}) - S$, entonces $\mathcal{P}$ se detiene partiendo desde $\|x_1, ...x_n, \alpha_1, ..., \alpha_m\|$ y la variable N1 queda con contenido igual a 0

Si $\mathcal{P}$ es un programa el cual computa a $\chi_S^{\omega^n \times \Sigma^{*m}}$, diremos que $\mathcal{P}$ *decide la pertenecia a* S, con respecto al conjunto $\omega^n \times \Sigma^{*m}$.

Macros asociados a conjuntos Σ-computables La proposicion anterior nos dice que si $S \subseteq \omega^n \times \Sigma^{*m}$ es un conjunto Σ-computable, entonces, ya que $\chi_S^{\omega^n \times \Sigma^{*m}}$ es Σ-computable, hay un macro

$$\left[\mathrm{IF}\ \chi_S^{\omega^n \times \Sigma^{*m}}(\mathrm{V1}, ..., \mathrm{V}\bar{n}, \mathrm{W1}, ..., \mathrm{W}\bar{m})\ \mathrm{GOTO\ A1}\right]$$

Escribiremos el nombre de este macro de la siguiente manera mas intuitiva:

$$[\text{IF } (\text{V}1, ..., \text{V}\bar{n}, \text{W}1, ..., \text{W}\bar{m}) \in S \text{ GOTO A}1]$$

Notese que las expanciones de este macro, dado que $\chi_S^{\omega^n \times \Sigma^{*m}}$ es Σ-total, ya sea terminan por la ultima instruccion de la expansion o direccionan a la primera instruccion que tenga label igual al label que reemplazo a A1 en la expansion. Es importante notar que para asegurar la existencia de este macro utilizamos que S es Σ-computable lo cual no siempre sucedera para un conjunto S. Por ejemplo, puede pasar que S sea el dominio de una funcion Σ-computable pero que S no sea Σ-computable (esto se vera mas adelante) y en tal caso no existira un macro

$$[\text{IF } (\text{V}1, ..., \text{V}\bar{n}, \text{W}1, ..., \text{W}\bar{m}) \in S \text{ GOTO A}1]$$

ya que si tal macro existiera seria facil hacer un programa que compute a $\chi_S^{\omega^n \times \Sigma^{*m}}$ y S seria Σ-computable. Es muy comun el error de suponer que existe un macro $[\text{IF } (\text{V}1, ..., \text{V}\bar{n}, \text{W}1, ..., \text{W}\bar{m}) \in S \text{ GOTO A}1]$ cuando S es el dominio de una funcion Σ-computable.

3.4 Batallas entre paradigmas

En esta seccion compararemos los tres paradigmas de computabilidad efectiva que hemos desarrollado anteriormente. Para esto probaremos que cada uno de dichos paradigmas "vence" al otro en el sentido que incluye por lo menos todas las funciones que incluye el otro en su modelizacion del concepto de funcion Σ-efectivamente computable.

3.4.1 Neumann vence a Godel

Usando macros podemos ahora probar que el paradigma imperativo de Neumann es por lo menos tan abarcativo como el funcional de Godel. Mas concretamente:

Theorem 90 *Si h es Σ-recursiva, entonces h es Σ-computable.*

Proof. Probaremos por induccion en k que

(*) Si $h \in \text{R}_k^\Sigma$, entonces h es Σ-computable.

El caso $k = 0$ es dejado al lector. Supongamos (*) vale para k, veremos que vale para $k + 1$. Sea $h \in \text{R}_{k+1}^\Sigma - \text{R}_k^\Sigma$. Hay varios casos

Caso 1. Supongamos $h = M(P)$, con $P : \omega \times \omega^n \times \Sigma^{*m} \to \omega$, un predicado perteneciente a R_k^Σ. Por hipotesis inductiva, P es Σ-computable y por lo tanto tenemos un macro

$$\left[\text{IF } P(\text{V}1, ..., \text{V}\overline{n+1}, \text{W}1, ..., \text{W}\bar{m}) \text{ GOTO A}1\right]$$

lo cual nos permite realizar el siguiente programa

$$\begin{array}{ll} \mathrm{L}2 & \left[\mathrm{IF}\ P(\mathrm{N}\overline{n+1},\mathrm{N}1,...,\mathrm{N}\bar{n},\mathrm{P}1,...,\mathrm{P}\bar{m})\ \mathrm{GOTO\ L1}\right] \\ & \mathrm{N}\overline{n+1} \leftarrow \mathrm{N}\overline{n+1}+1 \\ & \mathrm{GOTO\ L2} \\ \mathrm{L}1 & \mathrm{N}1 \leftarrow \mathrm{N}\overline{n+1} \end{array}$$

Es facil chequear que este programa computa h.

Caso 2. Supongamos $h = R(f,\mathcal{G})$, con

$$\begin{array}{rcl} f & : & S_1 \times ... \times S_n \times L_1 \times ... \times L_m \rightarrow \Sigma^* \\ \mathcal{G}_a & : & S_1 \times ... \times S_n \times L_1 \times ... \times L_m \times \Sigma^* \times \Sigma^* \rightarrow \Sigma^*,\ a \in \Sigma \end{array}$$

elementos de R_k^{Σ}. Sea $\Sigma = \{a_1,...,a_r\}$. Por hipotesis inductiva, las funciones f, $\mathcal{G}_a$, $a \in \Sigma$, son Σ-computables y por lo tanto podemos hacer el siguiente programa via el uso de macros

$$\begin{array}{rl} & \left[\mathrm{P}\overline{m+3} \leftarrow f(\mathrm{N}1,...,\mathrm{N}\bar{n},\mathrm{P}1,...,\mathrm{P}\bar{m})\right] \\ \mathrm{L}\overline{r+1} & \mathrm{IF\ P}\overline{m+1}\ \mathrm{BEGINS}\ a_1\ \mathrm{GOTO\ L1} \\ & \vdots \\ & \mathrm{IF\ P}\overline{m+1}\ \mathrm{BEGINS}\ a_r\ \mathrm{GOTO\ L}\bar{r} \\ & \mathrm{GOTO\ L}\overline{r+2} \\ \mathrm{L}1 & \mathrm{P}\overline{m+1} \leftarrow {}^{\frown}\mathrm{P}\overline{m+1} \\ & \left[\mathrm{P}\overline{m+3} \leftarrow \mathcal{G}_{a_1}(\mathrm{N}1,...,\mathrm{N}\bar{n},\mathrm{P}1,...,\mathrm{P}\bar{m},\mathrm{P}\overline{m+2},\mathrm{P}\overline{m+3})\right] \\ & \mathrm{P}\overline{m+2} \leftarrow \mathrm{P}\overline{m+2}.a_1 \\ & \mathrm{GOTO\ L}\overline{r+1} \\ & \vdots \\ \mathrm{L}\bar{r} & \mathrm{P}\overline{m+1} \leftarrow {}^{\frown}\mathrm{P}\overline{m+1} \\ & \left[\mathrm{P}\overline{m+3} \leftarrow \mathcal{G}_{a_r}(\mathrm{N}1,...,\mathrm{N}\bar{n},\mathrm{P}1,...,\mathrm{P}\bar{m},\mathrm{P}\overline{m+2},\mathrm{P}\overline{m+3})\right] \\ & \mathrm{P}\overline{m+2} \leftarrow \mathrm{P}\overline{m+2}.a_r \\ & \mathrm{GOTO\ L}\overline{r+1} \\ \mathrm{L}\overline{r+2} & \mathrm{P}1 \leftarrow \mathrm{P}\overline{m+3} \end{array}$$

Es facil chequear que este programa computa h.

El resto de los casos son dejados al lector. ■

Corollary 91 *Si*

$$\begin{array}{rcl} f & : & D_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow \omega \\ g & : & D_g \subseteq \omega^n \times \Sigma^{*m} \rightarrow \Sigma^* \\ P & : & D_P \subseteq \omega^n \times \Sigma^{*m} \rightarrow \{0,1\} \end{array}$$

son Σ*-recursivas, entonces hay macros*

$$\left[\mathrm{V}\overline{n+1} \leftarrow f(\mathrm{V}1,...,\mathrm{V}\bar{n},\mathrm{W}1,...,\mathrm{W}\bar{m})\right]$$
$$\left[\mathrm{W}\overline{m+1} \leftarrow g(\mathrm{V}1,...,\mathrm{V}\bar{n},\mathrm{W}1,...,\mathrm{W}\bar{m})\right]$$
$$[\mathrm{IF}\ P(\mathrm{V}1,...,\mathrm{V}\bar{n},\mathrm{W}1,...,\mathrm{W}\bar{m})\ \mathrm{GOTO\ A1}]$$

Se lleno de macros Cabe destacar que el corolario anterior nos dice que hay macros

$$\begin{array}{l}\left[\mathrm{V}\overline{n+1} \leftarrow f(\mathrm{V1},...,\mathrm{V}\bar{n},\mathrm{W1},...,\mathrm{W}\bar{m})\right] \\ \left[\mathrm{W}\overline{m+1} \leftarrow g(\mathrm{V1},...,\mathrm{V}\bar{n},\mathrm{W1},...,\mathrm{W}\bar{m})\right] \\ [\mathrm{IF}\ P(\mathrm{V1},...,\mathrm{V}\bar{n},\mathrm{W1},...,\mathrm{W}\bar{m})\ \mathrm{GOTO\ A1}]\end{array}$$

para todas las funciones Σ-mixtas y predicados Σ-mixtos que hemos trabajado hasta el momento en la materia ya que todas eran Σ-p.r.. Esto transforma al lenguaje $\mathcal{S}^\Sigma$ en un potente y relativamente comodo lenguaje de programacion ya que ahora tenemos macros para todas las funciones y predicados cotidianos en la matematica. Por ejemplo a continuacion usaremos la existencia de los macros [IF V1 es par GOTO A1] y [V2 $\leftarrow$ $\lfloor$V1/2$\rfloor$] para probar el siguiente resultado cuya prueba esta inspirada en su analoga del paradigma de computabilidad efectiva.

Lemma 92 *Supongamos $S_1, S_2 \subseteq \omega^n \times \Sigma^{*m}$ son conjuntos Σ-enumerables. Entonces $S_1 \cup S_2$ es Σ-enumerable.*

Proof. Podemos suponer que ni S_1 ni S_2 son vacios ya que de lo contrario los resultados son triviales. Ademas supondremos que $n = 2$ y $m = 1$.

La idea de la prueba es la misma que la que usamos para probar que la union de conjuntos Σ-efectivamente enumerables es Σ-efectivamente enumerable. Daremos usando macros un programa que enumera a $S_1 \cup S_2$ y luego aplicaremos la Proposicion 89. Por hipotesis hay funciones $F : \omega \to \omega \times \omega \times \Sigma^*$ y $G : \omega \to \omega \times \omega \times \Sigma^*$ tales que $F_{(1)}$, $F_{(2)}$, $F_{(3)}$, $G_{(1)}$, $G_{(2)}$ y $G_{(3)}$ son Σ-computables, $\mathrm{Im}(F) = S_1$ y $\mathrm{Im}(G) = S_2$. O sea que hay macros

$$\begin{array}{c}\left[\mathrm{V2} \leftarrow F_{(1)}(\mathrm{V1})\right] \\ \left[\mathrm{V2} \leftarrow F_{(2)}(\mathrm{V1})\right] \\ \left[\mathrm{W1} \leftarrow F_{(3)}(\mathrm{V1})\right] \\ \left[\mathrm{V2} \leftarrow G_{(1)}(\mathrm{V1})\right] \\ \left[\mathrm{V2} \leftarrow G_{(2)}(\mathrm{V1})\right] \\ \left[\mathrm{W1} \leftarrow G_{(3)}(\mathrm{V1})\right]\end{array}$$

Ya que el predicado $Par = \lambda x[x \text{ es par}]$ es Σ-p.r., el Corolario 91 nos dice que hay un macro:

$$[\mathrm{IF}\ Par(\mathrm{V1})\ \mathrm{GOTO\ A1}]$$

el cual escribiremos de la siguiente manera mas intuitiva

$$[\mathrm{IF\ V1\ es\ par\ GOTO\ A1}]$$

Ya que la funcion $D = \lambda x[\lfloor x/2 \rfloor]$ es Σ-p.r., el Corolario 91 nos dice que hay un macro:

$$[\mathrm{V2} \leftarrow D(\mathrm{V1})]$$

el cual escribiremos de la siguiente manera mas intuitiva

$$[\mathrm{V2} \leftarrow \lfloor \mathrm{V1}/2 \rfloor]$$

Sea $\mathcal{P}$ el siguiente programa:

$$\begin{array}{ll}
 & [\text{IF N1 es par GOTO L1} \\
 & \mathrm{N1} \leftarrow \mathrm{N1} \dot{-} 1 \\
 & [\mathrm{N1111} \leftarrow \lfloor \mathrm{N1}/2 \rfloor] \\
 & \left[\mathrm{N1} \leftarrow G_{(1)}(\mathrm{N1111})\right] \\
 & \left[\mathrm{N2} \leftarrow G_{(2)}(\mathrm{N1111})\right] \\
 & \left[\mathrm{P1} \leftarrow G_{(3)}(\mathrm{N1111})\right] \\
 & \text{GOTO L2} \\
\mathrm{L1} & [\mathrm{N1111} \leftarrow \lfloor \mathrm{N1}/2 \rfloor] \\
 & \left[\mathrm{N1} \leftarrow F_{(1)}(\mathrm{N1111})\right] \\
 & \left[\mathrm{N2} \leftarrow F_{(2)}(\mathrm{N1111})\right] \\
 & \left[\mathrm{P1} \leftarrow F_{(3)}(\mathrm{N1111})\right] \\
\mathrm{L2} & \text{SKIP}
\end{array}$$

Es facil ver que $\mathcal{P}$ cumple a y b de (3) de la Proposicion 89 por lo cual $S_1 \cup S_2$ es Σ-enumerable. ■

Tal como se vio en este ejemplo, el Corolario 91 junto con nuestra gran coleccion de funciones ya probadamente Σ-recursivas, nos permite simular con programas muchos de los procedimientos efectivos realizados anteriormente. Mas capacidad de simulacion obtendremos luego de ver que Godel vence a Neumann ya que la equivalencia de estos dos paradigmas nos asegura la existencia de macros que permitiran dentro de un programa hablar acerca del funcionamiento de otro programa. Esto sera clave a la hora de simular con programas a procedimientos efectivos que en su funcionamiento involucran el funcionamiento de otros procedimientos.

3.4.2 Godel vence a Neumann

Para probar que toda funcion Σ-computable es Σ-recursiva debemos hacer un profundo estudio de la recursividad del lenguaje $\mathcal{S}^\Sigma$. Primero analizaremos la recursividad de la sintaxis de $\mathcal{S}^\Sigma$.

Analisis de la recursividad de la sintaxis de $\mathcal{S}^\Sigma$ Primero probaremos dos lemas que muestran que la sintaxis de $\mathcal{S}^\Sigma$ es $(\Sigma \cup \Sigma_p)$-recursiva primitiva.

Recordemos que $S : Num^* \to Num^*$ fue definida de la siguiente manera

$$\begin{array}{rcl} S(\varepsilon) & = & 1 \\ S(\alpha 0) & = & \alpha 1 \\ S(\alpha 1) & = & \alpha 2 \\ S(\alpha 2) & = & \alpha 3 \\ S(\alpha 3) & = & \alpha 4 \\ S(\alpha 4) & = & \alpha 5 \\ S(\alpha 5) & = & \alpha 6 \\ S(\alpha 6) & = & \alpha 7 \\ S(\alpha 7) & = & \alpha 8 \\ S(\alpha 8) & = & \alpha 9 \\ S(\alpha 9) & = & S(\alpha)0 \end{array}$$

Tambien $\bar{\_} : \omega \to Num^*$ fue definida de la siguiente manera

$$\begin{array}{rcl} \bar{0} & = & \varepsilon \\ \overline{n+1} & = & S(\bar{n}) \end{array}$$

Es obvio de las definiciones que ambas funciones son Num-p.r.. Mas aun tenemos

Lemma 93 *Sea Σ un alfabeto cualquiera. Las funciones S y $\bar{\_}$ son $(\Sigma \cup \Sigma_p)$-p.r..*

Proof. Es facil ver que S y $\bar{\_}$ son Num-p.r.. Ya que tambien son $(\Sigma \cup \Sigma_p)$-mixtas, el Teorema 84 nos dice que ambas son $(\Sigma \cup \Sigma_p)$-p.r.. ■

Recordemos que $Bas : \mathrm{Ins}^\Sigma \to (\Sigma \cup \Sigma_p)^*$, fue definida por

$$Bas(I) = \begin{cases} J & \text{si } I \text{ es de la forma } \mathrm{L}\bar{k}J \text{ con } J \in \mathrm{Ins}^\Sigma \\ I & \text{caso contrario} \end{cases}$$

Definamos $Lab : \mathrm{Ins}^\Sigma \to (\Sigma \cup \Sigma_p)^*$ de la siguiente manera

$$Lab(I) = \begin{cases} \mathrm{L}\bar{k} & \text{si } I \text{ es de la forma } \mathrm{L}\bar{k}J \text{ con } J \in \mathrm{Ins}^\Sigma \\ \varepsilon & \text{caso contrario} \end{cases}$$

Lemma 94 *Para cada $n, x \in \omega$, tenemos que $|\bar{n}| \leq x$ si y solo si $n \leq 10^x - 1$*

Lemma 95 Ins^Σ *es un conjunto $(\Sigma \cup \Sigma_p)$-p.r..*

Proof. Para simplificar la prueba asumiremos que $\Sigma = \{@, \blacktriangle\}$. Ya que Ins^{Σ} es union de los siguientes conjuntos

$$\begin{array}{rcl}
L_1 & = & \left\{\mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{k}+1 : k \in \mathbf{N}\right\} \\
L_2 & = & \left\{\mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{k}\dot{-}1 : k \in \mathbf{N}\right\} \\
L_3 & = & \left\{\mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{n} : k, n \in \mathbf{N}\right\} \\
L_4 & = & \left\{\mathrm{N}\bar{k} \leftarrow 0 : k \in \mathbf{N}\right\} \\
L_5 & = & \left\{\mathrm{IF\ N}\bar{k} \neq 0 \mathrm{\ GOTO\ L}\bar{m} : k, m \in \mathbf{N}\right\} \\
L_6 & = & \left\{\mathrm{P}\bar{k} \leftarrow \mathrm{P}\bar{k}.@ : k \in \mathbf{N}\right\} \\
L_7 & = & \left\{\mathrm{P}\bar{k} \leftarrow \mathrm{P}\bar{k}.\blacktriangle : k \in \mathbf{N}\right\} \\
L_8 & = & \left\{\mathrm{P}\bar{k} \leftarrow {}^{\frown}\mathrm{P}\bar{k} : k \in \mathbf{N}\right\} \\
L_9 & = & \left\{\mathrm{P}\bar{k} \leftarrow \mathrm{P}\bar{n} : k, n \in \mathbf{N}\right\} \\
L_{10} & = & \left\{\mathrm{P}\bar{k} \leftarrow \varepsilon : k \in \mathbf{N}\right\} \\
L_{11} & = & \left\{\mathrm{IF\ P}\bar{k} \mathrm{\ BEGINS\ @\ GOTO\ L}\bar{m} : k, m \in \mathbf{N}\right\} \\
L_{12} & = & \left\{\mathrm{IF\ P}\bar{k} \mathrm{\ BEGINS\ } \blacktriangle \mathrm{\ GOTO\ L}\bar{m} : k, m \in \mathbf{N}\right\} \\
L_{13} & = & \{\mathrm{GOTO\ L}\bar{m} : m \in \mathbf{N}\} \\
L_{14} & = & \{\mathrm{SKIP}\} \\
L_{15} & = & \left\{\mathrm{L}\bar{k}\alpha : k \in \mathbf{N} \text{ y } \alpha \in L_1 \cup ... \cup L_{14}\right\}
\end{array}$$

solo debemos probar que $L_1, ..., L_{15}$ son $(\Sigma \cup \Sigma_p)$-p.r.. Veremos primero por ejemplo que

$$L_{11} = \left\{\mathrm{IFP}\bar{k}\mathrm{BEGINS@GOTOL}\bar{m} : k, m \in \mathbf{N}\right\}$$

es $(\Sigma \cup \Sigma_p)$-p.r.. Primero notese que $\alpha \in L_{11}$ si y solo si existen $k, m \in \mathbf{N}$ tales que

$$\alpha = \mathrm{IFP}\bar{k}\mathrm{BEGINS@GOTOL}\bar{m}$$

Mas formalmente tenemos que $\alpha \in L_{11}$ si y solo si

$$(\exists k \in \mathbf{N})(\exists m \in \mathbf{N})\ \alpha = \mathrm{IFP}\bar{k}\mathrm{BEGINS@GOTOL}\bar{m}$$

Ya que cuando existen tales k, m tenemos que $\bar{k}$ y $\bar{m}$ son subpalabras de α, el lema anterior nos dice que $\alpha \in L_{11}$ si y solo si

$$(\exists k \in \mathbf{N})_{k \leq 10^{|\alpha|}}(\exists m \in \mathbf{N})_{m \leq 10^{|\alpha|}}\ \alpha = \mathrm{IFP}\bar{k}\mathrm{BEGINS@GOTOL}\bar{m}$$

Sea

$$P = \lambda mk\alpha\left[\alpha = \mathrm{IFP}\bar{k}\mathrm{BEGINS@GOTOL}\bar{m}\right]$$

Ya que $D_{\lambda k\left[\bar{k}\right]} = \omega$, tenemos que $D_P = \omega^2 \times (\Sigma \cup \Sigma_p)^*$. Notese que

$$P = \lambda\alpha\beta\left[\alpha = \beta\right] \circ \left[p_3^{2,1}, f\right]$$

donde

$$f=\lambda\alpha_1\alpha_2\alpha_3\alpha_4\left[\alpha_1\alpha_2\alpha_3\alpha_4\right]\circ\left[C^{2,1}_{\text{IFP}},\lambda k\left[\bar{k}\right]\circ p^{2,1}_2,C^{2,1}_{\text{BEGINS@GOTOL}},\lambda k\left[\bar{k}\right]\circ p^{2,1}_1\right]$$

lo cual nos dice que P es $(\Sigma\cup\Sigma_p)$-p.r..

Notese que

$$\chi^{(\Sigma\cup\Sigma_p)^*}_{L_{11}}=\lambda\alpha\left[(\exists k\in\mathbf{N})_{k\leq 10^{|\alpha|}}(\exists m\in\mathbf{N})_{m\leq 10^{|\alpha|}}\ P(m,k,\alpha)\right]$$

Esto nos dice que podemos usar dos veces el Lema 62 para ver que $\chi^{(\Sigma\cup\Sigma_p)^*}_{L_{11}}$ es $(\Sigma\cup\Sigma_p)$-p.r.. Veamos como. Sea

$$Q=\lambda k\alpha\left[(\exists m\in\mathbf{N})_{m\leq 10^{|\alpha|}}\ P(m,k,\alpha)\right]$$

Por el Lema 62 tenemos que

$$\lambda xk\alpha\left[(\exists m\in\mathbf{N})_{m\leq x}\ P(m,k,\alpha)\right]$$

es $(\Sigma\cup\Sigma_p)$-p.r. lo cual nos dice que

$$Q=\lambda xk\alpha\left[(\exists m\in\mathbf{N})_{m\leq x}\ P(m,k,\alpha)\right]\circ\left[\lambda\alpha\left[10^{|\alpha|}\right]\circ p^{1,1}_2,p^{1,1}_1,p^{1,1}_2\right]$$

lo es. Ya que

$$\chi^{(\Sigma\cup\Sigma_p)^*}_{L_{11}}=\lambda\alpha\left[(\exists k\in\mathbf{N})_{k\leq 10^{|\alpha|}}\ Q(k,\alpha)\right]$$

podemos en forma similar aplicar el Lema 62 y obtener finalmente que $\chi^{(\Sigma\cup\Sigma_p)^*}_{L_{11}}$ es $(\Sigma\cup\Sigma_p)$-p.r..

En forma similar podemos probar que $L_1,...,L_{14}$ son $(\Sigma\cup\Sigma_p)$-p.r.. Esto nos dice que $L_1\cup...\cup L_{14}$ es $(\Sigma\cup\Sigma_p)$-p.r.. Notese que $L_1\cup...\cup L_{14}$ es el conjunto de las instrucciones basicas de $\mathcal{S}^\Sigma$. Llamemos InsBas^Σ a dicho conjunto. Para ver que L_{15} es $(\Sigma\cup\Sigma_p)$-p.r. notemos que

$$\chi^{(\Sigma\cup\Sigma_p)^*}_{L_{15}}=\lambda\alpha\left[(\exists k\in\mathbf{N})_{k\leq 10^{|\alpha|}}(\exists\beta\in\text{InsBas}^\Sigma)_{|\beta|\leq|\alpha|}\ \alpha=\text{L}\bar{k}\beta\right]$$

lo cual nos dice que aplicando dos veces el Lema 62 obtenemos que $\chi^{(\Sigma\cup\Sigma_p)^*}_{L_{15}}$ es $(\Sigma\cup\Sigma_p)$-p.r.. Ya que $\text{Ins}^\Sigma=\text{InsBas}^\Sigma\cup L_{15}$ tenemos que Ins^Σ es $(\Sigma\cup\Sigma_p)$-p.r..
■

@@finpagina@@

Lemma 96 *Bas y Lab son funciones* $(\Sigma\cup\Sigma_p)$*-p.r.*

Proof. Sea $\leq$ un orden total sobre $\Sigma\cup\Sigma_p$. Sea $L=\{\text{L}\bar{k}:k\in\mathbf{N}\}\cup\{\varepsilon\}$. Dejamos al lector probar que L es un conjunto $(\Sigma\cup\Sigma_p)$-p.r.. Sea

$$P=\lambda I\alpha\left[\alpha\in\text{Ins}^\Sigma\wedge I\in\text{Ins}^\Sigma\wedge[\alpha]_1\neq\text{L}\wedge(\exists\beta\in L)\ I=\beta\alpha\right]$$

Note que $D_P=(\Sigma\cup\Sigma_p)^{*2}$. Dejamos al lector probar que P es $(\Sigma\cup\Sigma_p)$-p.r.. Notese ademas que cuando $I\in\text{Ins}^\Sigma$ tenemos que $P(I,\alpha)=1$ sii $\alpha=Bas(I)$.

Dejamos al lector probar que $Bas = M^{\leq}(P)$ por lo que para ver que Bas es $(\Sigma \cup \Sigma_p)$-p.r., solo nos falta ver que la funcion Bas es acotada por alguna funcion $(\Sigma \cup \Sigma_p)$-p.r. y $(\Sigma \cup \Sigma_p)$-total. Pero esto es trivial ya que $|Bas(I)| \leq |I| = \lambda\alpha[|\alpha|](I)$, para cada $I \in \mathrm{Ins}^{\Sigma}$.

Finalmente note que

$$Lab = M^{\leq}(\lambda I\alpha\,[\alpha Bas(I) = I])$$

lo cual nos dice que Lab es $(\Sigma \cup \Sigma_p)$-p.r.. ■

Recordemos que dado un programa $\mathcal{P}$ habiamos definido $I_i^{\mathcal{P}} = \varepsilon$, para $i = 0$ o $i > n(\mathcal{P})$. O sea que la funcion $(\Sigma \cup \Sigma_p)$-mixta $\lambda i\mathcal{P}\left[I_i^{\mathcal{P}}\right]$ tiene dominio igual a $\omega \times \mathrm{Pro}^{\Sigma}$. Notese que usamos notacion lambda respecto del alfabeto $\Sigma \cup \Sigma_p$. Ademas notese que usamos la variable $\mathcal{P}$ en la notacion lambda por un tema de comodidad psicologica dado que la expresion I_i^{α} esta definida solo cuando α es un programa pero podriamos haber escrito $\lambda i\alpha\,[I_i^{\alpha}]$ y sigue siendo la misma funcion.

Lemma 97 *(a)* Pro^{Σ} *es un conjunto* $(\Sigma \cup \Sigma_p)$*-p.r.*

(b) $\lambda\mathcal{P}\,[n(\mathcal{P})]$ *y* $\lambda i\mathcal{P}\left[I_i^{\mathcal{P}}\right]$ *son funciones* $(\Sigma \cup \Sigma_p)$*-p.r..*

Proof. Ya que $\mathrm{Pro}^{\Sigma} = D_{\lambda\mathcal{P}[n(\mathcal{P})]}$ tenemos que (b) implica (a). Para probar (b) Sea $\leq$ un orden total sobre $\Sigma \cup \Sigma_p$. Sea P el siguiente predicado

$$\begin{aligned}\lambda x\,\big[&Lt(x) > 0 \wedge (\forall t \in \mathbf{N})_{t \leq Lt(x)}\ *^{\leq}((x)_t) \in \mathrm{Ins}^{\Sigma} \wedge \\ &(\forall t \in \mathbf{N})_{t \leq Lt(x)}(\forall m \in \mathbf{N})\ \neg(\mathrm{L}\bar{m}\text{ t-final } *^{\leq}((x)_t)) \vee \\ &(\exists j \in \mathbf{N})_{j \leq Lt(x)}(\exists \alpha \in (\Sigma \cup \Sigma_p) - Num)\ \mathrm{L}\bar{m}\alpha\text{ t-inicial } *^{\leq}((x)_j)\big]\end{aligned}$$

Notese que $D_P = \mathbf{N}$ y que $P(x) = 1$ sii $Lt(x) > 0$, $*^{\leq}((x)_t) \in \mathrm{Ins}^{\Sigma}$, para cada $t = 1, ..., Lt(x)$ y ademas $\subset_{t=1}^{t=Lt(x)} *^{\leq}((x)_t) \in \mathrm{Pro}^{\Sigma}$. Para ver que P es $(\Sigma \cup \Sigma_p)$-p.r. solo nos falta acotar el cuantificador $(\forall m \in \mathbf{N})$ de la expresion lambda que define a P. Ya que nos interesan los valores de m para los cuales $\bar{m}$ es posiblemente una subpalabra de alguna de las palabras $*^{\leq}((x)_j)$, el Lema 94 nos dice que una cota posible es $10^{\max\{|*^{\leq}((x)_j)|:1 \leq j \leq Lt(x)\}} - 1$. Dejamos al lector los detalles de la prueba de que P es $(\Sigma \cup \Sigma_p)$-p.r.. Sea

$$Q = \lambda x\alpha\left[P(x) \wedge \alpha = \subset_{t=1}^{t=Lt(x)} *^{\leq}((x)_t)\right].$$

Note que $D_Q = \mathbf{N} \times (\Sigma \cup \Sigma_p)^*$. Claramente Q es $(\Sigma \cup \Sigma_p)$-p.r.. Ademas note que $D_{M(Q)} = \mathrm{Pro}^{\Sigma}$. Notese que para $\mathcal{P} \in \mathrm{Pro}^{\Sigma}$, tenemos que $M(Q)(\mathcal{P})$ es aquel numero tal que pensado como infinitupla (via mirar su secuencia de exponentes) codifica la secuencia de instrucciones que forman a $\mathcal{P}$. Es decir

$$M(Q)(\mathcal{P}) = \left\langle \#^{\leq}(I_1^{\mathcal{P}}), \#^{\leq}(I_2^{\mathcal{P}}), ..., \#^{\leq}(I_{n(\mathcal{P})}^{\mathcal{P}}), 0, 0, ... \right\rangle$$

Por (b) del Lema 67, $M(Q)$ es $(\Sigma\cup\Sigma_p)$-p.r. ya que para cada $\mathcal{P}\in\mathrm{Pro}^\Sigma$ tenemos que

$$\begin{array}{rcl} M(Q)(\mathcal{P}) & = & \left\langle \#^{\leq}(I_1^{\mathcal{P}}), \#^{\leq}(I_2^{\mathcal{P}}), ..., \#^{\leq}(I_{n(\mathcal{P})}^{\mathcal{P}}), 0, 0, ... \right\rangle \\ & = & \prod\limits_{i=1}^{n(\mathcal{P})} pr(i)^{\#^{\leq}(I_1^{\mathcal{P}})} \\ & \leq & \prod\limits_{i=1}^{|\mathcal{P}|} pr(i)^{\#^{\leq}(\mathcal{P})} \end{array}$$

Ademas tenemos que

$$\begin{array}{rcl} \lambda\mathcal{P}\left[n(\mathcal{P})\right] & = & \lambda x\left[Lt(x)\right]\circ M(Q) \\ \lambda i\mathcal{P}\left[I_i^{\mathcal{P}}\right] & = & *^{\leq}\circ g\circ\left[p_1^{1,1}, M(Q)\circ p_2^{1,1}\right] \end{array}$$

donde $g = C_0^{1,1}|_{\{0\}\times\omega}\cup\lambda ix\left[(x)_i\right]$, lo cual dice que $\lambda\mathcal{P}\left[n(\mathcal{P})\right]$ y $\lambda i\mathcal{P}\left[I_i^{\mathcal{P}}\right]$ son funciones $(\Sigma\cup\Sigma_p)$-p.r.. ■

Analisis de la recursividad de la semantica de $\mathcal{S}^\Sigma$ Para estudiar la recursividad de la semantica de $\mathcal{S}^\Sigma$ deberemos definir varias funciones que tienen que ver con el funcionamiento de un programa y estudiar su recursividad.

Las funciones $i^{n,m}$, $E_{\#}^{n,m}$ y $E_*^{n,m}$ Sean $n,m\geq 0$ fijos. Definamos entonces las funciones

$$\begin{array}{rcl} i^{n,m} & : & \omega\times\omega^n\times\Sigma^{*m}\times\mathrm{Pro}^\Sigma\to\omega \\ E_{\#}^{n,m} & : & \omega\times\omega^n\times\Sigma^{*m}\times\mathrm{Pro}^\Sigma\to\omega^{[\mathbf{N}]} \\ E_*^{n,m} & : & \omega\times\omega^n\times\Sigma^{*m}\times\mathrm{Pro}^\Sigma\to\Sigma^{*[\mathbf{N}]} \end{array}$$

de la siguiente manera

$$\begin{array}{rcl} (i^{n,m}(0,\vec{x},\vec{\alpha},\mathcal{P}), E_{\#}^{n,m}(0,\vec{x},\vec{\alpha},\mathcal{P}), E_*^{n,m}(0,\vec{x},\vec{\alpha},\mathcal{P})) & = & (1,(x_1,...,x_n,0,...),(\alpha_1,...,\alpha_m,\varepsilon,...)) \\ (i^{n,m}(t+1,\vec{x},\vec{\alpha},\mathcal{P}), E_{\#}^{n,m}(t+1,\vec{x},\vec{\alpha},\mathcal{P}), E_*^{n,m}(t+1,\vec{x},\vec{\alpha},\mathcal{P})) & = & S_{\mathcal{P}}(i^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), E_{\#}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), E_*^{n,m}(t \end{array}$$

Notese que

$$(i^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), E_{\#}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), E_*^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}))$$

es la descripcion instantanea que se obtiene luego de correr $\mathcal{P}$ una cantidad t de pasos partiendo del estado

$$((x_1,...,x_n,0,...),(\alpha_1,...,\alpha_m,\varepsilon,...))$$

Es importante notar que si bien $i^{n,m}$ es una funcion $(\Sigma\cup\Sigma_p)$-mixta, ni $E_{\#}^{n,m}$ ni $E_{*}^{n,m}$ lo son.

Definamos para cada $j\in\mathbf{N}$, funciones

$$\begin{array}{rcl} E_{\# j}^{n,m} & : & \omega\times\omega^n\times\Sigma^{*m}\times\mathrm{Pro}^{\Sigma}\rightarrow\omega \\ E_{* j}^{n,m} & : & \omega\times\omega^n\times\Sigma^{*m}\times\mathrm{Pro}^{\Sigma}\rightarrow\Sigma^{*} \end{array}$$

de la siguiente manera

$$\begin{array}{rcl} E_{\# j}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}) & = & j\text{-esima coordenada de } E_{\#}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}) \\ E_{* j}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}) & = & j\text{-esima coordenada de } E_{*}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}) \end{array}$$

Notese que

$$\begin{array}{rcl} E_{\#}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}) & = & (E_{\# 1}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}),E_{\# 2}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}),...) \\ E_{*}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}) & = & (E_{* 1}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}),E_{* 2}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}),...) \end{array}$$

Nuestro proximo objetivo es mostrar que las funciones $i^{n,m}$, $E_{\# j}^{n,m}$, $E_{* j}^{n,m}$ son $(\Sigma\cup\Sigma_p)$-p.r.

Para esto primero debemos probar un lema el cual muestre que una ves codificadas las descripciones instantaneas en forma numerica, las funciones que dan la descripcion instantanea sucesora son $(\Sigma\cup\Sigma_p)$-p.r.. Dado un orden total $\leq$ sobre $\Sigma\cup\Sigma_p$, codificaremos las descripciones instantaneas haciendo uso de las biyecciones

$$\begin{array}{rcl} \omega^{[\mathbf{N}]} & \rightarrow & \mathbf{N} \\ (s_1,s_2,...) & \rightarrow & \langle s_1,s_2,...\rangle \end{array} \qquad \begin{array}{rcl} \Sigma^{*[\mathbf{N}]} & \rightarrow & \mathbf{N} \\ (\sigma_1,\sigma_2,...) & \rightarrow & \left\langle \#^{\leq}(\sigma_1),\#^{\leq}(\sigma_2),...\right\rangle \end{array}$$

Es decir que a la descripcion instantanea

$$(i,(s_1,s_2,...),(\sigma_1,\sigma_2,...))$$

la codificaremos con la terna

$$(i,\langle s_1,s_2,...\rangle,\left\langle \#^{\leq}(\sigma_1),\#^{\leq}(\sigma_2),...\right\rangle)\in\omega\times\mathbf{N}\times\mathbf{N}$$

Es decir que una terna $(i,x,y)\in\omega\times\mathbf{N}\times\mathbf{N}$ codificara a la descripcion instantanea

$$(i,((x)_1,(x)_2,...),(*^{\leq}((y)_1),*^{\leq}((y)_2),...))$$

Definamos

$$\begin{array}{rcl} s & : & \omega\times\mathbf{N}\times\mathbf{N}\times\mathrm{Pro}^{\Sigma}\rightarrow\omega \\ S_{\#} & : & \omega\times\mathbf{N}\times\mathbf{N}\times\mathrm{Pro}^{\Sigma}\rightarrow\omega \\ S_{*} & : & \omega\times\mathbf{N}\times\mathbf{N}\times\mathrm{Pro}^{\Sigma}\rightarrow\omega \end{array}$$

de la siguiente manera

$$s(i,x,y,\mathcal{P}) = \begin{array}{l}\text{primera coordenada de la codificacion de la descripcion instantanea} \\ \text{sucesora de } (i,((x)_1,(x)_2,...),(*^{\leq}((y)_1),*^{\leq}((y)_2),...)) \text{ en } \mathcal{P}\end{array}$$

$$S_{\#}(i,x,y,\mathcal{P}) = \begin{array}{l}\text{segunda coordenada de la codificacion de la descripcion instantanea} \\ \text{sucesora de } (i,((x)_1,(x)_2,...),(*^{\leq}((y)_1),*^{\leq}((y)_2),...)) \text{ en } \mathcal{P}\end{array}$$

$$S_{*}(i,x,y,\mathcal{P}) = \begin{array}{l}\text{tercera coordenada de la codificacion de la descripcion instantanea} \\ \text{sucesora de } (i,((x)_1,(x)_2,...),(*^{\leq}((y)_1),*^{\leq}((y)_2),...)) \text{ en } \mathcal{P}\end{array}$$

Notese que la definicion de estas funciones depende del orden total $\leq$ sobre $\Sigma \cup \Sigma_p$.

@@finpagina@@

Lemma 98 *Dado un orden total $\leq$ sobre $\Sigma \cup \Sigma_p$, las funciones s, $S_{\#}$ y S_* son $(\Sigma \cup \Sigma_p)$-p.r..*

Proof. Necesitaremos algunas funciones $(\Sigma \cup \Sigma_p)$-p.r.. Dada una instruccion I en la cual al menos ocurre una variable, usaremos $\#Var1(I)$ para denotar el numero de la primer variable que ocurre en I. Por ejemplo

$$\#Var1\left(\mathrm{L}\bar{n} \text{ IF } \mathrm{N}\bar{k} \neq 0 \text{ GOTO } \mathrm{L}\bar{m}\right) = k$$

Notese que $\lambda I[\#Var1(I)]$ tiene dominio igual a $\mathrm{Ins}^{\Sigma} - L$, donde L es la union de los siguientes conjuntos

$$\begin{array}{c}\{\text{GOTO L}\bar{m} : m \in \mathbf{N}\} \cup \{\mathrm{L}\bar{k} \text{ GOTO L}\bar{m} : k,m \in \mathbf{N}\} \\ \{\text{SKIP}\} \cup \{\mathrm{L}\bar{k} \text{ SKIP} : k \in \mathbf{N}\}\end{array}$$

Dada una instruccion I en la cual ocurren dos variables, usaremos $\#Var2(I)$ para denotar el numero de la segunda variable que ocurre en I. Por ejemplo

$$\#Var2\left(\mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{m}\right) = m$$

Notese que el dominio de $\lambda I[\#Var2(I)]$ es igual a la union de los siguientes conjuntos

$$\begin{array}{lcl}\{\mathrm{N}\bar{k} & \leftarrow & \mathrm{N}\bar{m} : k,m \in \mathbf{N}\} \cup \{\mathrm{L}\bar{j}\ \mathrm{N}\bar{k} \leftarrow \mathrm{N}\bar{m} : j,k,m \in \mathbf{N}\} \\ \{\mathrm{P}\bar{k} & \leftarrow & \mathrm{P}\bar{m} : k,m \in \mathbf{N}\} \cup \{\mathrm{L}\bar{j}\ \mathrm{P}\bar{k} \leftarrow \mathrm{P}\bar{m} : j,k,m \in \mathbf{N}\}\end{array}$$

Ademas notese que para una instruccion I tenemos que

$$\begin{array}{lcl}\#Var1(I) & = & \min\limits_k(\mathrm{N}\bar{k}\leftarrow \text{ ocu } I \vee \mathrm{N}\bar{k}\neq \text{ ocu } I \vee \mathrm{P}\bar{k}\leftarrow \text{ ocu } I \vee \mathrm{P}\bar{k}\mathrm{B} \text{ ocu } I) \\ \#Var2(I) & = & \min\limits_k(\mathrm{N}\bar{k} \text{ t-final } I \vee \mathrm{N}\bar{k}+ \text{ ocu } I \vee \mathrm{N}\bar{k}\dot{-} \text{ ocu } I \vee \mathrm{P}\bar{k} \text{ t-final } I \vee \mathrm{P}\bar{k}. \text{ ocu } I)\end{array}$$

Esto nos dice que si llamamos P al predicado

$$\lambda k\alpha\left[\alpha\in\mathrm{Ins}^{\Sigma}\wedge(\mathrm{N}\bar{k}{\leftarrow}\text{ ocu }\alpha\vee\mathrm{N}\bar{k}{\neq}\text{ ocu }\alpha\vee\mathrm{P}\bar{k}{\leftarrow}\text{ ocu }\alpha\vee\mathrm{P}\bar{k}\mathrm{B}\text{ ocu }\alpha)\right]$$

entonces $\lambda I[\#Var1(I)] = M(P)$ por lo cual $\lambda I[\#Var1(I)]$ es $(\Sigma\cup\Sigma_p)$-p.r. Similarmente se puede ver que $\lambda I[\#Var2(I)]$ es $(\Sigma\cup\Sigma_p)$-p.r.. Sea

$$\begin{array}{rcl} F_{\dot{-}}:\mathbf{N}\times\mathbf{N} & \rightarrow & \omega \\ (x,j) & \rightarrow & \left\langle (x)_1,....,(x)_{j-1},(x)_j\dot{-}1,(x)_{j+1},...\right\rangle \end{array}$$

Ya que

$$F_{\dot{-}}(x,j)=\left\{\begin{array}{ll} Q(x,pr(j)) & \text{si } pr(j) \text{ divide } x \\ x & \text{caso contrario}\end{array}\right.$$

tenemos que $F_{\dot{-}}$ es $(\Sigma\cup\Sigma_p)$-p.r.. Sea

$$\begin{array}{rcl} F_{+}:\mathbf{N}\times\mathbf{N} & \rightarrow & \omega \\ (x,j) & \rightarrow & \langle (x)_1,....,(x)_{j-1},(x)_j+1,(x)_{j+1},...\rangle \end{array}$$

Ya que $F_{+}(x,j)=x.pr(j)$ tenemos que F_{+} es $(\Sigma\cup\Sigma_p)$-p.r.. Sea

$$\begin{array}{rcl} F_{\leftarrow}:\mathbf{N}\times\mathbf{N}\times\mathbf{N} & \rightarrow & \omega \\ (x,j,k) & \rightarrow & \langle (x)_1,....,(x)_{j-1},(x)_k,(x)_{j+1},...\rangle \end{array}$$

Ya que $F_{\leftarrow}(x,j,k)=Q(x,pr(j)^{(x)_j}).pr(j)^{(x)_k}$ tenemos que $F_{\leftarrow}$ es $(\Sigma\cup\Sigma_p)$-p.r.. Sea

$$\begin{array}{rcl} F_{0}:\mathbf{N}\times\mathbf{N} & \rightarrow & \omega \\ (x,j) & \rightarrow & \langle (x)_1,....,(x)_{j-1},0,(x)_{j+1},...\rangle \end{array}$$

Es facil ver que F_0 es $(\Sigma\cup\Sigma_p)$-p.r.. Para cada $a\in\Sigma$, sea

$$\begin{array}{rcl} F_{a}:\mathbf{N}\times\mathbf{N} & \rightarrow & \omega \\ (x,j) & \rightarrow & \left\langle (x)_1,....,(x)_{j-1},\#^{\leq}(*^{\leq}((x)_j)a),(x)_{j+1},...\right\rangle \end{array}$$

Es facil ver que F_a es $(\Sigma\cup\Sigma_p)$-p.r.. En forma similar puede ser probado que

$$\begin{array}{rcl} F_{\curvearrowright}:\mathbf{N}\times\mathbf{N} & \rightarrow & \omega \\ (x,j) & \rightarrow & \left\langle (x)_1,....,(x)_{j-1},\#^{\leq}(\curvearrowright(*^{\leq}((x)_j))),(x)_{j+1},...\right\rangle \end{array}$$

es $(\Sigma\cup\Sigma_p)$-p.r.

Dado $(i,x,y,\mathcal{P})\in\omega\times\mathbf{N}\times\mathbf{N}\times\mathrm{Pro}^{\Sigma}$, tenemos varios casos en los cuales los valores $s(i,x,y,\mathcal{P}),S_{\#}(i,x,y,\mathcal{P})$ y $S_{*}(i,x,y,\mathcal{P})$ pueden ser obtenidos usando las funciones antes definidas:

(1) CASO $i=0\vee i>n(\mathcal{P})$. Entonces

$$\begin{array}{rcl} s(i,x,y,\mathcal{P}) & = & i \\ S_{\#}(i,x,y,\mathcal{P}) & = & x \\ S_{*}(i,x,y,\mathcal{P}) & = & y \end{array}$$

(2) CASO $(\exists j \in \omega)\ Bas(I_i^{\mathcal{P}}) = \mathrm{N}\bar{j} \leftarrow \mathrm{N}\bar{j} + 1$. Entonces

$$\begin{array}{rcl} s(i,x,y,\mathcal{P}) & = & i+1 \\ S_{\#}(i,x,y,\mathcal{P}) & = & F_{+}(x, \#Var1(I_i^{\mathcal{P}})) \\ S_{*}(i,x,y,\mathcal{P}) & = & y \end{array}$$

(3) CASO $(\exists j \in \omega)\ Bas(I_i^{\mathcal{P}}) = \mathrm{N}\bar{j} \leftarrow \mathrm{N}\bar{j} \dot{-} 1$. Entonces

$$\begin{array}{rcl} s(i,x,y,\mathcal{P}) & = & i+1 \\ S_{\#}(i,x,y,\mathcal{P}) & = & F_{\dot{-}}(x, \#Var1(I_i^{\mathcal{P}})) \\ S_{*}(i,x,y,\mathcal{P}) & = & y \end{array}$$

(4) CASO $(\exists j,k \in \omega)\ Bas(I_i^{\mathcal{P}}) = \mathrm{N}\bar{j} \leftarrow \mathrm{N}\bar{k}$. Entonces

$$\begin{array}{rcl} s(i,x,y,\mathcal{P}) & = & i+1 \\ S_{\#}(i,x,y,\mathcal{P}) & = & F_{\leftarrow}(x, \#Var1(I_i^{\mathcal{P}}), \#Var2(I_i^{\mathcal{P}})) \\ S_{*}(i,x,y,\mathcal{P}) & = & y \end{array}$$

(5) CASO $(\exists j,k \in \omega)\ Bas(I_i^{\mathcal{P}}) = \mathrm{N}\bar{j} \leftarrow 0$. Entonces

$$\begin{array}{rcl} s(i,x,y,\mathcal{P}) & = & i+1 \\ S_{\#}(i,x,y,\mathcal{P}) & = & F_{0}(x, \#Var1(I_i^{\mathcal{P}})) \\ S_{*}(i,x,y,\mathcal{P}) & = & y \end{array}$$

(6) CASO $(\exists j,m \in \omega)\ \left(Bas(I_i^{\mathcal{P}}) = \mathrm{IF\ N}\bar{j} \neq 0 \mathrm{\ GOTO\ L}\bar{m} \wedge (x)_j = 0\right)$. Entonces

$$\begin{array}{rcl} s(i,x,y,\mathcal{P}) & = & i+1 \\ S_{\#}(i,x,y,\mathcal{P}) & = & x \\ S_{*}(i,x,y,\mathcal{P}) & = & y \end{array}$$

(7) CASO $(\exists j,m \in \omega)\ \left(Bas(I_i^{\mathcal{P}}) = \mathrm{IF\ N}\bar{j} \neq 0 \mathrm{\ GOTO\ L}\bar{m} \wedge (x)_j \neq 0\right)$. Entonces

$$\begin{array}{rcl} s(i,x,y,\mathcal{P}) & = & \min_l \left(Lab(I_l^{\mathcal{P}}) \neq \varepsilon \wedge Lab(I_l^{\mathcal{P}}) \text{ t-final } I_i^{\mathcal{P}}\right) \\ S_{\#}(i,x,y,\mathcal{P}) & = & x \\ S_{*}(i,x,y,\mathcal{P}) & = & y \end{array}$$

(8) CASO $(\exists j \in \omega)\ Bas(I_i^{\mathcal{P}}) = \mathrm{P}\bar{j} \leftarrow \mathrm{P}\bar{j}.a$. Entonces

$$\begin{array}{rcl} s(i,x,y,\mathcal{P}) & = & i+1 \\ S_{\#}(i,x,y,\mathcal{P}) & = & x \\ S_{*}(i,x,y,\mathcal{P}) & = & F_{a}(y, \#Var1(I_i^{\mathcal{P}})) \end{array}$$

(9) CASO $(\exists j \in \omega)\ Bas(I_i^{\mathcal{P}}) = \mathrm{P}\bar{j} \leftarrow {}^\frown\mathrm{P}\bar{j}$. Entonces

$$\begin{array}{rcl} s(i,x,y,\mathcal{P}) & = & i+1 \\ S_{\#}(i,x,y,\mathcal{P}) & = & x \\ S_{*}(i,x,y,\mathcal{P}) & = & F_{\frown}(y, \#Var1(I_i^{\mathcal{P}})) \end{array}$$

(10) CASO $(\exists j,k \in \omega)\ Bas(I_i^{\mathcal{P}}) = \mathrm{P}\bar{j} \leftarrow \mathrm{P}\bar{k}$. Entonces

$$\begin{array}{rcl} s(i,x,y,\mathcal{P}) & = & i+1 \\ S_{\#}(i,x,y,\mathcal{P}) & = & x \\ S_{*}(i,x,y,\mathcal{P}) & = & F_{\leftarrow}(y, \#Var1(I_i^{\mathcal{P}}), \#Var2(I_i^{\mathcal{P}})) \end{array}$$

(11) CASO $(\exists j \in \omega)\ Bas(I_i^{\mathcal{P}}) = \mathrm{P}\bar{j} \leftarrow \varepsilon$. Entonces

$$\begin{array}{rcl} s(i,x,y,\mathcal{P}) & = & i+1 \\ S_{\#}(i,x,y,\mathcal{P}) & = & x \\ S_{*}(i,x,y,\mathcal{P}) & = & F_{0}(y, \#Var1(I_i^{\mathcal{P}})) \end{array}$$

(12) CASO $(\exists j,m \in \omega)(\exists a \in \Sigma)\ \left(Bas(I_i^{\mathcal{P}}) = \mathrm{IF\ P}\bar{j}\ \mathrm{BEGINS}\ a\ \mathrm{GOTO\ L}\bar{m} \wedge [*^{\leq}((y)_j)]_1 \neq a\right)$. Entonces

$$\begin{array}{rcl} s(i,x,y,\mathcal{P}) & = & i+1 \\ S_{\#}(i,x,y,\mathcal{P}) & = & x \\ S_{*}(i,x,y,\mathcal{P}) & = & y \end{array}$$

(13) CASO $(\exists j,m \in \omega)(\exists a \in \Sigma)\ \left(Bas(I_i^{\mathcal{P}}) = \mathrm{IF\ P}\bar{j}\ \mathrm{BEGINS}\ a\ \mathrm{GOTO\ L}\bar{m} \wedge [*^{\leq}((y)_j)]_1 = a\right)$. Entonces

$$\begin{array}{rcl} s(i,x,y,\mathcal{P}) & = & \min_l \left(Lab(I_l^{\mathcal{P}}) \neq \varepsilon \wedge Lab(I_l^{\mathcal{P}}) \text{ t-final } I_i^{\mathcal{P}}\right) \\ S_{\#}(i,x,y,\mathcal{P}) & = & x \\ S_{*}(i,x,y,\mathcal{P}) & = & y \end{array}$$

(14) CASO $(\exists j \in \omega)\ Bas(I_i^{\mathcal{P}}) = \mathrm{GOTO\ L}\bar{j}$. Entonces

$$\begin{array}{rcl} s(i,x,y,\mathcal{P}) & = & \min_l \left(Lab(I_l^{\mathcal{P}}) \neq \varepsilon \wedge Lab(I_l^{\mathcal{P}}) \text{ t-final } I_i^{\mathcal{P}}\right) \\ S_{\#}(i,x,y,\mathcal{P}) & = & x \\ S_{*}(i,x,y,\mathcal{P}) & = & y \end{array}$$

(15) CASO $Bas(I_i^{\mathcal{P}}) = \mathrm{SKIP}$. Entonces

$$\begin{array}{rcl} s(i,x,y,\mathcal{P}) & = & i+1 \\ S_{\#}(i,x,y,\mathcal{P}) & = & x \\ S_{*}(i,x,y,\mathcal{P}) & = & y \end{array}$$

O sea que los casos anteriores nos permiten definir conjuntos $S_1,...,S_{15}$, los cuales son disjuntos de a pares y cuya union da el conjunto $\omega\times\mathbf{N}\times\mathbf{N}\times\mathrm{Pro}^{\Sigma}$, de manera que cada una de las funciones $s,S_{\#}$ y S_{*} pueden escribirse como union disjunta de funciones $(\Sigma\cup\Sigma_p)$-p.r. restrinjidas respectivamente a los conjuntos $S_1,...,S_{15}$. Ya que los conjuntos $S_1,...,S_{15}$ son $(\Sigma\cup\Sigma_p)$-p.r. el Lema 58 nos dice que $s,S_{\#}$ y S_{*} lo son. ■

Aparte del lema anterior, para probar que las funciones $i^{n,m}$, $E_{\#j}^{n,m}$ y $E_{*j}^{n,m}$ son $(\Sigma\cup\Sigma_p)$-p.r., nos sera necesario el siguiente resultado. Recordemos que para $x_1,...,x_n\in\omega$, usabamos $\langle x_1,...,x_n\rangle$ para denotar $\langle x_1,...,x_n,0,...\rangle$. Ademas recordemos que en el Lema 70 probamos que para cada $n\geq 1$, la funcion $\lambda x_1...x_n\left[\langle x_1,...,x_n\rangle\right]$ es $\emptyset$-p.r.

Lemma 99 *Sean*

$$\begin{array}{rcl} f_i & : & S_1\times...\times S_n\times L_1\times...\times L_m\rightarrow\omega \\ g_i & : & \omega^k\times\omega\times S_1\times...\times S_n\times L_1\times...\times L_m\rightarrow\omega \\ F_i & : & \omega\times S_1\times...\times S_n\times L_1\times...\times L_m\rightarrow\omega \end{array}$$

con $i=1,...,k$, funciones Σ-mixtas. Supongamos que

$$\begin{array}{rcl} F_i(0,\vec{x},\vec{\alpha}) & = & f_i(0,\vec{x},\vec{\alpha}) \\ F_i(t+1,\vec{x},\vec{\alpha}) & = & g_i(F_1(t,\vec{x},\vec{\alpha}),...,F_k(t,\vec{x},\vec{\alpha}),t,\vec{x},\vec{\alpha}) \end{array}$$

para cada $i=1,...,k$, $t\in\omega$ y $(\vec{x},\vec{\alpha})\in S_1\times...\times S_n\times L_1\times...\times L_m$. Entonces si las funciones $f_1,...,f_k,g_1,...,g_k$ son Σ-p.r., las funciones $F_1,...,F_k$ lo son.

Proof. Para mayor claridad haremos el caso $k=2$. Sea

$$F=\lambda t\vec{x}\vec{\alpha}\left[\langle F_1(t,\vec{x},\vec{\alpha}),F_2(t,\vec{x},\vec{\alpha})\rangle\right]$$

Es claro que si F es Σ-p.r., entonces lo es cada F_i. Notese que

$$\begin{array}{rcl} F(0,\vec{x},\vec{\alpha}) & = & \langle f_1(\vec{x},\vec{\alpha}),f_2(\vec{x},\vec{\alpha})\rangle \\ F(t+1,\vec{x},\vec{\alpha}) & = & \langle g_1((F(t,\vec{x},\vec{\alpha}))_1,(F(t,\vec{x},\vec{\alpha}))_2,t,\vec{x},\vec{\alpha}),g_2((F(t,\vec{x},\vec{\alpha}))_1,(F(t,\vec{x},\vec{\alpha}))_2,t,\vec{x},\vec{\alpha})\rangle \end{array}$$

lo cual nos dice que $F=R(f,g)$ donde

$$\begin{array}{rcl} f & = & \lambda\vec{x}\vec{\alpha}\left[\langle f_1(\vec{x},\vec{\alpha}),f_2(\vec{x},\vec{\alpha})\rangle\right] \\ g & = & \lambda At\vec{x}\vec{\alpha}\left[\langle g_1((A)_1,(A)_2,t,\vec{x},\vec{\alpha}),g_2((A)_1,(A)_2,t,\vec{x},\vec{\alpha})\rangle\right] \end{array}$$

■

@@finpagina@@

Ahora usando los dos lemas anteriores podemos probar el siguiente importante resultado.

Proposition 100 *Sean $n,m\geq 0$. Las funciones $i^{n,m}$, $E_{\#j}^{n,m}$, $E_{*j}^{n,m}$, $j=1,2,...$, son $(\Sigma\cup\Sigma_p)$-p.r.*

Proof. Sea $\leq$ un orden total sobre $\Sigma \cup \Sigma_p$ y sean s, $S_{\#}$ y S_* las funciones definidas previamente al Lema 98. Definamos

$$\begin{array}{rcl} K_{\#}^{n,m} & = & \lambda t\vec{x}\vec{\alpha}\mathcal{P}\left[\left\langle E_{\#1}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), E_{\#2}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), ...\right\rangle\right] \\ K_{*}^{n,m} & = & \lambda t\vec{x}\vec{\alpha}\mathcal{P}\left[\left\langle \#^{\leq}(E_{*1}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P})), \#^{\leq}(E_{*2}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P})), ...\right\rangle\right] \end{array}$$

Notese que

$$\begin{array}{rcl} i^{n,m}(0,\vec{x},\vec{\alpha},\mathcal{P}) & = & 1 \\ K_{\#}^{n,m}(0,\vec{x},\vec{\alpha},\mathcal{P}) & = & \langle x_1,...,x_n\rangle \\ K_{*}^{n,m}(0,\vec{x},\vec{\alpha},\mathcal{P}) & = & \left\langle \#^{\leq}(\alpha_1),...,\#^{\leq}(\alpha_m)\right\rangle \\ i^{n,m}(t+1,\vec{x},\vec{\alpha},\mathcal{P}) & = & s(i^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), K_{\#}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), K_{*}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P})) \\ K_{\#}^{n,m}(t+1,\vec{x},\vec{\alpha},\mathcal{P}) & = & S_{\#}(i^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), K_{\#}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), K_{*}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P})) \\ K_{*}^{n,m}(t+1,\vec{x},\vec{\alpha},\mathcal{P}) & = & S_{*}(i^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), K_{\#}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), K_{*}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P})) \end{array}$$

Por el Lema 99 tenemos que $i^{n,m}$, $K_{\#}^{n,m}$ y $K_{*}^{n,m}$ son $(\Sigma \cup \Sigma_p)$-p.r.. Ademas notese que

$$\begin{array}{rcl} E_{\#j}^{n,m} & = & \lambda t\vec{x}\vec{\alpha}\mathcal{P}\left[(K_{\#}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}))_j\right] \\ E_{*j}^{n,m} & = & \lambda t\vec{x}\vec{\alpha}\mathcal{P}\left[*^{\leq}((K_{*}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}))_j)\right] \end{array}$$

por lo cual las funciones $E_{\#j}^{n,m}$, $E_{*j}^{n,m}$, $j = 1, 2, ...$, son $(\Sigma \cup \Sigma_p)$-p.r. ■

Las funciones $Halt^{n,m}$ y $T^{n,m}$ Dados $n, m \in \omega$, definamos:

$$Halt^{n,m} = \lambda t\vec{x}\vec{\alpha}\mathcal{P}\left[i^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}) = n(\mathcal{P}) + 1\right]$$

Notese que $D_{Halt^{n,m}} = \omega \times \omega^n \times \Sigma^{*m} \times \mathrm{Pro}^{\Sigma}$ (ojo que aqui la notacion lambda es respecto del alfabeto $\Sigma \cup \Sigma_p$). Ademas notese que usamos la variable $\mathcal{P}$ en la notacion lambda por un tema de comodidad psicologica dado que $i^{n,m}$ esta definida solo cuando la ultima coordenada es un programa pero podriamos haber escrito $\lambda t\vec{x}\vec{\alpha}\alpha\left[i^{n,m}(t,\vec{x},\vec{\alpha},\alpha) = n(\alpha) + 1\right]$ y sigue siendo la misma funcion.

Cabe destacar que $Halt^{n,m}$ tiene una descripcion muy intuitiva, ya que dado $(t,\vec{x},\vec{\alpha},\mathcal{P}) \in \omega \times \omega^n \times \Sigma^{*m} \times \mathrm{Pro}^{\Sigma}$, tenemos que $Halt^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}) = 1$ si y solo si el programa $\mathcal{P}$ se detiene luego de t pasos partiendo desde el estado $\|x_1,...,x_n,\alpha_1,...,\alpha_m\|$.

Lemma 101 *$Halt^{n,m}$ es $(\Sigma \cup \Sigma_p)$-p.r.*

Proof. Notar que $Halt^{n,m} = \lambda xy[x = y] \circ \left[i^{n,m}, \lambda\mathcal{P}[n(\mathcal{P}) + 1] \circ p_{1+n+m+1}^{1+n,m+1}\right]$. ■

Ahora definamos $T^{n,m} = M(Halt^{n,m})$. Notese que

$$D_{T^{n,m}} = \{(\vec{x},\vec{\alpha},\mathcal{P}) : \mathcal{P} \text{ se detiene partiendo de } \|x_1,...,x_n,\alpha_1,...,\alpha_m\|\}$$

y para $(\vec{x},\vec{\alpha},\mathcal{P})\in D_{T^{n,m}}$ tenemos que $T^{n,m}(\vec{x},\vec{\alpha},\mathcal{P})=$ cantidad de pasos necesarios para que $\mathcal{P}$ se detenga partiendo de $\|x_1,...,x_n,\alpha_1,...,\alpha_m\|$. En algun sentido, la funcion $T^{n,m}$ mide el tiempo que tarda en detenerse $\mathcal{P}$

Proposition 102 *$T^{n,m}$ es $(\Sigma\cup\Sigma_p)$-recursiva*

Proof. Es directo del Lema 67 ya que $Halt^{n,m}$ es $(\Sigma\cup\Sigma_p)$-p.r. ■

Las funciones $\Phi_{\#}^{n,m}$ y $\Phi_{*}^{n,m}$ Para $n,m\in\omega$ definamos la funcion $\Phi_{\#}^{n,m}$ de la siguiente manera:

$$\begin{aligned} D_{\Phi_{\#}^{n,m}} &= \left\{(\vec{x},\vec{\alpha},\mathcal{P})\in\omega^n\times\Sigma^{*m}\times\mathrm{Pro}^{\Sigma}:(\vec{x},\vec{\alpha})\in D_{\Psi_{\mathcal{P}}^{n,m,\#}}\right\} \\ \Phi_{\#}^{n,m}(\vec{x},\vec{\alpha},\mathcal{P}) &= \Psi_{\mathcal{P}}^{n,m,\#}(\vec{x},\vec{\alpha}),\text{ para cada }(\vec{x},\vec{\alpha},\mathcal{P})\in D_{\Phi_{\#}^{n,m}} \end{aligned}$$

Notese que

$$D_{\Phi_{\#}^{n,m}}=\{(\vec{x},\vec{\alpha},\mathcal{P}):\mathcal{P}\text{ se detiene partiendo de }\|x_1,...,x_n,\alpha_1,...,\alpha_m\|\}$$

y para cada $(\vec{x},\vec{\alpha},\mathcal{P})\in D_{\Phi_{\#}^{n,m}}$, se tiene que $\Phi_{\#}^{n,m}(\vec{x},\vec{\alpha},\mathcal{P})=$ valor que queda en la variable N1 cuando $\mathcal{P}$ se detiene partiendo de $\|x_1,...,x_n,\alpha_1,...,\alpha_m\|$.

Similarmente, definamos la funcion $\Phi_{*}^{n,m}$ de la siguiente manera:

$$\begin{aligned} D_{\Phi_{*}^{n,m}} &= \left\{(\vec{x},\vec{\alpha},\mathcal{P})\in\omega^n\times\Sigma^{*m}\times\mathrm{Pro}^{\Sigma}:(\vec{x},\vec{\alpha})\in D_{\Psi_{\mathcal{P}}^{n,m,*}}\right\} \\ \Phi_{*}^{n,m}(\vec{x},\vec{\alpha},\mathcal{P}) &= \Psi_{\mathcal{P}}^{n,m,*}(\vec{x},\vec{\alpha}),\text{ para cada }(\vec{x},\vec{\alpha},\mathcal{P})\in D_{\Phi_{*}^{n,m}} \end{aligned}$$

Notese que

$$\begin{aligned} \Phi_{\#}^{n,m} &= \lambda\vec{x}\vec{\alpha}\mathcal{P}\left[\Psi_{\mathcal{P}}^{n,m,\#}(\vec{x},\vec{\alpha})\right] \\ \Phi_{*}^{n,m} &= \lambda\vec{x}\vec{\alpha}\mathcal{P}\left[\Psi_{\mathcal{P}}^{n,m,*}(\vec{x},\vec{\alpha})\right] \end{aligned}$$

Theorem 103 *Las funciones $\Phi_{\#}^{n,m}$ y $\Phi_{*}^{n,m}$ son $(\Sigma\cup\Sigma_p)$-recursivas.*

Proof. Veremos que $\Phi_{\#}^{n,m}$ es $(\Sigma\cup\Sigma_p)$-recursiva. Notar que $D_{T^{n,m}}=D_{\Phi_{\#}^{n,m}}$. Notese que para $(\vec{x},\vec{\alpha},\mathcal{P})\in D_{T^{n,m}}=D_{\Phi_{\#}^{n,m}}$ tenemos que

$$\Phi_{\#}^{n,m}(\vec{x},\vec{\alpha},\mathcal{P})=E_{\#1}^{n,m}\left(T^{n,m}(\vec{x},\vec{\alpha},\mathcal{P}),\vec{x},\vec{\alpha},\mathcal{P}\right)$$

lo cual con un poco mas de trabajo nos permite probar que

$$\Phi_{\#}^{n,m}=E_{\#1}^{n,m}\circ\left[T^{n,m},p_1^{n,m+1},...,p_{n+m+1}^{n,m+1}\right]$$

Ya que la funcion $E_{\#1}^{n,m}$ es $(\Sigma\cup\Sigma_p)$-p.r. y $T^{n,m}$ es $(\Sigma\cup\Sigma_p)$-r., tenemos que $\Phi_{\#}^{n,m}$ es $(\Sigma\cup\Sigma_p)$-r. ■

Godel vence a Neumann Ahora nos sera facil probar que el paradigma de Godel es por lo menos tan abarcativo como el imperativo de Von Neumann. Mas concretamente:

Theorem 104 *Si $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to O$ es Σ-computable, entonces f es Σ-recursiva.*

Proof. Haremos el caso $O = \Sigma^*$. Sea $\mathcal{P}_0$ un programa que compute a f. Primero veremos que f es $(\Sigma \cup \Sigma_p)$-recursiva. Note que

$$f = \Phi_*^{n,m} \circ \left[p_1^{n,m}, ..., p_{n+m}^{n,m}, C_{\mathcal{P}_0}^{n,m}\right]$$

donde cabe destacar que $p_1^{n,m}, ..., p_{n+m}^{n,m}$ son las proyecciones respecto del alfabeto $\Sigma \cup \Sigma_p$, es decir que tienen dominio $\omega^n \times (\Sigma \cup \Sigma_p)^{*m}$. Ya que $\Phi_*^{n,m}$ es $(\Sigma \cup \Sigma_p)$-recursiva tenemos que f lo es. O sea que el Teorema 84 nos dice que f es Σ-recursiva. ■

Un corolario interesante que se puede obtener del teorema anterior es que toda funcion Σ-recursiva puede obtenerse combinando las reglas basicas en una forma muy particular.

Corollary 105 *Si $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to O$ es Σ-recursiva, entonces existe un predicado Σ-p.r. $P : \mathbf{N} \times \omega^n \times \Sigma^{*m} \to \omega$ y una funcion Σ-p.r. $g : \mathbf{N} \to O$ tales que $f = g \circ M(P)$.*

Proof. Supongamos que $O = \Sigma^*$. Sea $\mathcal{P}_0$ un programa el cual compute a f. Sea $\leq$ un orden total sobre Σ. Note que podemos tomar

$$\begin{array}{rcl} P & = & \lambda t\vec{x}\vec{\alpha}[Halt^{n,m}((t)_1, \vec{x}, \vec{\alpha}, \mathcal{P}_0) \wedge (t)_2 = \#^{\leq}(E_{*1}^{n,m}((t)_1, \vec{x}, \vec{\alpha}, \mathcal{P}_0))] \\ g & = & \lambda t\left[*^{\leq}((t)_2)\right] \end{array}$$

(Justifique por que P es Σ-p.r..) ■

A continuacion veremos ejemplos naturales de funciones $(\Sigma \cup \Sigma_p)$-recursivas que no son $(\Sigma \cup \Sigma_p)$-recursivas primitivas. Cabe destacar que la prueba se basa en la Proposicion 66 (enunciada sin demostracion) la cual nos dice que cualquiera sea el alfabeto finito Σ, siempre hay una funcion que es Σ-recursiva y no es Σ-recursiva primitiva

Proposition 106 *Cualesquiera sean $n, m \in \omega$, se tiene que las funciones $T^{n,m}$, $\Phi_\#^{n,m}$ y $\Phi_*^{n,m}$ no son $(\Sigma \cup \Sigma_p)$-p.r.*

Proof. Fijemos $n, m \in \omega$. Probaremos que $\Phi_\#^{n,m}$ es $(\Sigma \cup \Sigma_p)$-p.r. sii $\Phi_\#^{0,0}$ es $(\Sigma \cup \Sigma_p)$-p.r.. Sean $f_1, f_2 : \omega^n \times \Sigma^{*m} \to (\Sigma \cup \Sigma_p)^*$ dadas por

$$\begin{array}{rcl} f_1(\vec{x}, \vec{\alpha}) & = & (\mathrm{N}1 \leftarrow \mathrm{N}1 + 1)^{x_1} ... (\mathrm{N}\bar{n} \leftarrow \mathrm{N}\bar{n} + 1)^{x_n} \\ f_1(\vec{x}, \vec{\alpha}) & = & \left(\subset_{i=1}^{i=|\alpha_1|} \mathrm{P}1 \leftarrow \mathrm{P}1.[\alpha_1]_i\right) ... \left(\subset_{i=1}^{i=|\alpha_m|} \mathrm{P}1 \leftarrow \mathrm{P}1.[\alpha_m]_i\right) \end{array}$$

Sea $f : \omega^n \times \Sigma^{*m} \times \text{Pro}^\Sigma \to (\Sigma \cup \Sigma_p)^*$ dada por

$$f(\vec{x}, \vec{\alpha}, \mathcal{P}) = f_1(\vec{x}, \vec{\alpha}) f_1(\vec{x}, \vec{\alpha}) \mathcal{P}$$

Es facil ver que f es $(\Sigma \cup \Sigma_p)$-p.r.. Notese que $\Phi_\#^{n,m} = \Phi_\#^{0,0} \circ f$. Ademas notese que

$$\Phi_\#^{0,0} = \Phi_\#^{n,m} \circ \left[C_0^{0,1}, ..., C_0^{0,1}, C_\varepsilon^{0,1}, ..., C_\varepsilon^{0,1}, p_1^{0,1}\right]$$

Ya que f y las funciones $C_0^{0,1}, C_\varepsilon^{0,1}, p_1^{0,1}$ son $(\Sigma \cup \Sigma_p)$-p.r., tenemos que $\Phi_\#^{n,m}$ es $(\Sigma \cup \Sigma_p)$-p.r. sii $\Phi_\#^{0,0}$ lo es.

Supongamos ahora que para algunos $k, l \in \omega$ se tiene que $\Phi_\#^{k,l}$ es $(\Sigma \cup \Sigma_p)$-p.r.. Llegaremos a un absurdo. Por lo antes probado tenemos que $\Phi_\#^{n,m}$ es $(\Sigma \cup \Sigma_p)$-p.r., cualesquiera sean $n, m \in \omega$. Notese que de la prueba del teorema anterior sigue que toda funcion Σ-computable con imagen contenida en ω es de la forma $\Phi_\#^{n,m} \circ \left[p_1^{n,m}, ..., p_{n+m}^{n,m}, C_{\mathcal{P}_0}^{n,m}\right]$, para algunos $n, m \in \omega$ y $\mathcal{P}_0 \in \text{Pro}^\Sigma$. Pero entonces toda funcion Σ-computable con imagen contenida en ω es $(\Sigma \cup \Sigma_p)$-p.r., lo cual por el Teorema 104 nos dice que toda funcion Σ-computable con imagen contenida en ω es $(\Sigma \cup \Sigma_p)$-p.r.. Esto contradice la Proposicion 66.

Ahora supongamos que hay $n, m \in \omega$ tales que $T^{n,m}$ es $(\Sigma \cup \Sigma_p)$-p.r.. Llegaremos a un absurdo. Como ya vimos en la prueba de un teorema reciente, se tiene que

$$\Phi_\#^{n,m} = E_{\#1}^{n,m} \circ \left[T^{n,m}, p_1^{n,m+1}, ..., p_{n+m+1}^{n,m+1}\right]$$

Pero entonces ya que $E_{\#1}^{n,m}$ es $(\Sigma \cup \Sigma_p)$-p.r., tenemos que $\Phi_\#^{n,m}$ es $(\Sigma \cup \Sigma_p)$-p.r., lo cual como ya vimos recien no es cierto. El absurdo nos dice que $T^{n,m}$ no es $(\Sigma \cup \Sigma_p)$-p.r.. ■

Corollary 107 *La minimizacion de un predicado Σ-p.r. no necesariamente es Σ-p.r.*

Proof. Por definicion $T^{n,m} = M(Halt^{n,m})$. ■

Uso de macros asociados a las funciones $Halt^{n,m}$, $E_\#^{n,m}$ y $E_*^{n,m}$ Aqui veremos, con ejemplos, como ciertos macros nos permitiran dentro de un programa hablar acerca del funcionamiento de otro programa. Esto junto con el hecho que cada funcion Σ-recursiva y cada predicado Σ-recursivo tienen su macro asociado (Corolario 91), sera muy util a la hora del diseño de programas y nos permitira simular dentro del paradigma imperativo muchas ideas usadas para el diseño de procedimientos efectivos. En este sentido la convinacion de los dos paradigmas (recursivo e imperativo) nos permite fortalecer notablemente al paradigma imperativo en su roll modelizador (o simulador) de los procedimientos efectivos. Esto es importante ya que el paradigma mas comodo, amplio e intuitivo, a la hora de decidir si algo es o no computable, es sin duda el filosofico o efectivo.

Veamos el primer ejemplo. Sea $\Sigma = \{@, !\}$ y sea $\mathcal{P}_0 \in \mathrm{Pro}^\Sigma$ tal que $0 \in \mathrm{Dom}\Psi^{1,0,\#}_{\mathcal{P}_0}$ y $\Psi^{1,0,\#}_{\mathcal{P}_0}(0) = 2$. Probaremos que

$$S = \{x \in \mathrm{Dom}\Psi^{1,0,\#}_{\mathcal{P}_0} : \Psi^{1,0,\#}_{\mathcal{P}_0}(x) \neq 0\}$$

es Σ-enumerable. Notese que $0 \in S$. Por definicion de conjunto Σ-enumerable, deberemos encontrar un programa $\mathcal{P} \in \mathrm{Pro}^\Sigma$ tal que $\mathrm{Dom}\Psi^{1,0,\#}_{\mathcal{P}} = \omega$ y $\mathrm{Im}\,\Psi^{1,0,\#}_{\mathcal{P}} = S$. Dicho en palabras, el programa $\mathcal{P}$ debera cumplir:

- siempre que lo corramos desde un estado de la forma $\|x\|$, con $x \in \omega$, debe detenerse y el contenido de la variable N1 bajo detencion debera ser un elemento de S
- para cada $s \in S$ debera haber un $x \in \omega$ tal que s es el valor de la variable N1 bajo detencion cuando corremos $\mathcal{P}$ desde $\|x\|$

A continuacion daremos una descripcion intuitiva del funcionamiento de $\mathcal{P}$ (pseudocodigo) para luego escribirlo correctamente usando macros. El programa $\mathcal{P}$ comenzara del estado $\|x\|$ y hara las siguientes tareas

Etapa 1: si $x = 0$ ir a Etapa 6, en caso contrario ir a Etapa 2.

Etapa 2: calcular $(x)_1$ y $(x)_2$ e ir a Etapa 3.

Etapa 3: si $\mathcal{P}_0$ termina desde $\|(x)_1\|$ en $(x)_2$ pasos ir a Etapa 4, en caso contrario ir a Etapa 6

Etapa 4: si el valor que queda en N1 luego de correr $\mathcal{P}_0$ una cantidad $(x)_2$ de pasos, partiendo de $\|(x)_1\|$, es distinto de 0, entonces ir a Etapa 5. En caso contrario ir a Etapa 6.

Etapa 5: asignar a N1 el valor $(x)_1$ y terminar

Etapa 6: asignar a N1 el valor 0 y terminar

Notese que la descripcion anterior no es ni mas ni menos que un procedimiento efectivo que enumera a S, y nuestra mision es simularlo dentro del lenguaje $\mathcal{S}^\Sigma$. Para esto usaremos varios macros. Ya que la funcion $f = \lambda x[(x)_1]$ es Σ-p.r., el Corolario 91 nos dice que hay un macro:

$$[\mathrm{V2} \leftarrow f(\mathrm{V1})]$$

el cual escribiremos de la siguiente manera mas intuitiva:

$$[\mathrm{V2} \leftarrow (\mathrm{V1})_1]$$

Similarmente hay un macro:

$$[\mathrm{V}2 \leftarrow (\mathrm{V}1)_2]$$

Tambien, ya que el predicado $P = \lambda x[x = 0]$ es Σ-recursivo, hay un macro:

$$[\mathrm{IF}\ P(\mathrm{V}1)\ \mathrm{GOTO\ A}1]$$

el cual escribiremos de la siguiente manera:

$$[\mathrm{IF\ V}1 = 0\ \mathrm{GOTO\ A}1]$$

Definamos

$$H = \lambda tx\left[Halt^{1,0}(t, x, \mathcal{P}_0)\right]$$

Notar que $D_H = \omega^2$ y que H es Σ-mixta. Ademas sabemos que la funcion $Halt^{1,0}$ es $(\Sigma \cup \Sigma_p)$-p.r. por lo cual resulta facilmente que H es $(\Sigma \cup \Sigma_p)$-p.r.. Por la Proposicion de Independencia del Alfabeto tenemos que H es Σ-p.r.. O sea que el Corolario 91 nos dice que hay un macro:

$$[\mathrm{IF}\ H(\mathrm{V}1, \mathrm{V}2)\ \mathrm{GOTO\ A}1]$$

Para hacer mas intuitivo el uso de este macro lo escribiremos de la siguiente manera

$$\left[\mathrm{IF}\ Halt^{1,0}(\mathrm{V}1, \mathrm{V}2, \mathcal{P}_0)\ \mathrm{GOTO\ A}1\right]$$

Sea

$$g = \lambda tx\left[E^{1,0}_{\#1}(t, x, \mathcal{P}_0)\right]$$

Ya que g es Σ-recursiva (por que?), hay un macro:

$$[\mathrm{V}3 \leftarrow g(\mathrm{V}1, \mathrm{V}2)]$$

Para hacer mas intuitivo el uso de este macro lo escribiremos de la siguiente manera

$$\left[\mathrm{V}3 \leftarrow E^{1,0}_{\#1}(\mathrm{V}1, \mathrm{V}2, \mathcal{P}_0)\right]$$

Ahora si podemos dar nuestro progama $\mathcal{P}$ que enumera a S:

$$\begin{array}{ll}
 & \mathrm{IF\ N}1 \neq 0\ \mathrm{GOTO\ L}1 \\
 & \mathrm{GOTO\ L}2 \\
\mathrm{L}1 & [\mathrm{N}3 \leftarrow (\mathrm{N}1)_1] \\
 & [\mathrm{N}4 \leftarrow (\mathrm{N}1)_2] \\
 & \left[\mathrm{IF}\ Halt^{1,0}(\mathrm{N}4, \mathrm{N}3, \mathcal{P}_0)\ \mathrm{GOTO\ L}3\right] \\
 & \mathrm{GOTO\ L}2 \\
\mathrm{L}3 & \left[\mathrm{N}5 \leftarrow E^{1,0}_{\#1}(\mathrm{N}4, \mathrm{N}3, \mathcal{P}_0)\right] \\
 & [\mathrm{IF\ N}5 = 0\ \mathrm{GOTO\ L}2] \\
 & \mathrm{N}1 \leftarrow \mathrm{N}3 \\
 & \mathrm{GOTO\ L}4 \\
\mathrm{L}2 & \mathrm{N}1 \leftarrow 0 \\
\mathrm{L}4 & \mathrm{SKIP}
\end{array}$$

Enumeracion de conjuntos de programas Ya que los programas de $\mathcal{S}^{\Sigma}$ son palabras del alfabeto $\Sigma \cup \Sigma_p$, nos podemos preguntar cuando un conjunto L de programas es $(\Sigma \cup \Sigma_p)$-enumerable. Daremos un ejemplo. Sea $\Sigma = \{@, !\}$ y sea

$$L = \{\mathcal{P} \in \mathrm{Pro}^{\Sigma} : \Psi_{\mathcal{P}}^{1,0,\#}(10) = 10\}$$

Veremos que L es $(\Sigma \cup \Sigma_p)$-enumerable, dando un programa $\mathcal{Q} \in \mathrm{Pro}^{\Sigma \cup \Sigma_p}$ que enumere a L, es decir tal que $\mathrm{Dom}(\Psi_{\mathcal{Q}}^{1,0,*}) = \omega$ y $\mathrm{Im}(\Psi_{\mathcal{Q}}^{1,0,*}) = L$. Cabe destacar que aqui hay en juego dos versiones de nuestro lenguaje imperativo, es decir enumeraremos un conjunto de programas de $\mathcal{S}^{\Sigma}$ usando un programa de $\mathcal{S}^{\Sigma \cup \Sigma_p}$. Sea $\leq$ un orden total sobre el conjunto $\Sigma \cup \Sigma_p$.

A continuacion daremos una descripcion intuitiva del funcionamiento de $\mathcal{Q}$ (pseudocodigo) para luego escribirlo correctamente usando macros. Notese que SKIP $\in L$. El programa $\mathcal{Q}$ comenzara del estado $\|x\|$ y hara las siguientes tareas

Etapa 1: si $x = 0$ ir a Etapa 6, en caso contrario ir a Etapa 2.

Etapa 2: calcular $(x)_1$, $(x)_2$ y $*^{\leq}((x)_1)$ e ir a Etapa 3.

Etapa 3: si $*^{\leq}((x)_1) \in \mathrm{Pro}^{\Sigma}$ y termina partiendo desde $\|10\|$ en $(x)_2$ pasos ir a Etapa 4, en caso contrario ir a Etapa 6

Etapa 4: si el valor que queda en N1 luego de correr $*^{\leq}((x)_1)$ una cantidad $(x)_2$ de pasos, partiendo de $\|10\|$ es igual a 10, entonces ir a Etapa 5. En caso contrario ir a Etapa 6.

Etapa 5: asignar a P1 la palabra $*^{\leq}((x)_1)$ y terminar

Etapa 6: asignar a P1 la palabra SKIP y terminar

Notese que la descripcion anterior no es ni mas ni menos que un procedimiento efectivo que enumera a L, y nuestra mision es simularlo dentro del lenguaje $\mathcal{S}^{\Sigma \cup \Sigma_p}$. Para esto usaremos varios macros. Es importante notar que los macros que usaremos corresponden al lenguaje $\mathcal{S}^{\Sigma \cup \Sigma_p}$ ya que los usaremos en $\mathcal{Q}$ el cual sera un programa de $\mathcal{S}^{\Sigma \cup \Sigma_p}$.

Ya que las funciones $\lambda x[(x)_1]$ y $\lambda x[(x)_2]$ son $(\Sigma \cup \Sigma_p)$-recursivas el Corolario 91 nos dice que hay macros asociados a estas funciones los cuales escribiremos de la siguiente manera mas intuitiva:

$$[\mathrm{V2} \leftarrow (\mathrm{V1})_1]$$
$$[\mathrm{V2} \leftarrow (\mathrm{V1})_2]$$

Ya que el predicado $P = \lambda x[x = 10]$ es $(\Sigma \cup \Sigma_p)$-recursivo tenemos su macro asociado el cual escribiremos de la siguiente manera:

$$[\mathrm{IF\ V1} = 10\ \mathrm{GOTO\ A1}]$$

Por un lema anterior sabemos que Pro^{Σ} es un conjunto $(\Sigma\cup\Sigma_p)$-p.r.´por lo cual $\chi^{(\Sigma\cup\Sigma_p)^*}_{\mathrm{Pro}^{\Sigma}}$ es $(\Sigma\cup\Sigma_p)$-p.r., por lo cual hay un macro

$$\left[\text{IF } \chi^{(\Sigma\cup\Sigma_p)^*}_{\mathrm{Pro}^{\Sigma}}(\mathrm{W1}) \text{ GOTO A1}\right]$$

el cual escribiremos de la siguiente manera

$$\left[\text{IF W1} \in \mathrm{Pro}^{\Sigma} \text{ GOTO A1}\right]$$

Ya que el predicado $Halt^{1,0}$ es $(\Sigma\cup\Sigma_p)$-recursivo tenemos un macro asociado a el, el cual escribiremos de la siguiente forma

$$\left[\text{IF } Halt^{1,0}(\mathrm{V1},\mathrm{V2},\mathrm{W1}) \text{ GOTO A1}\right]$$

Ya que $E^{1,0}_{\#1}$ es $(\Sigma\cup\Sigma_p)$-recursivo tenemos un macro asociado a ella, el cual escribiremos de la siguiente forma

$$\left[\mathrm{V3} \leftarrow E^{1,0}_{\#1}(\mathrm{V1},\mathrm{V2},\mathrm{W1})\right]$$

Tambien usaremos macros

$$[\mathrm{V1} \leftarrow 10]$$
$$[\mathrm{W1} \leftarrow \mathrm{SKIP}]$$

(dejamos al lector hacerlos a mano o tambien se puede justificar su existencia via la proposicion de existencia de macros aplicada a las funciones $C^{0,0}_{10}$ y $C^{0,0}_{\mathrm{SKIP}}$).

Ahora si podemos hacer el programa $\mathcal{Q}$ que enumera a L:

$$\begin{array}{ll}
 & \text{IF N1} \neq 0 \text{ GOTO L1} \\
 & \text{GOTO L2} \\
\text{L1} & [\mathrm{N2} \leftarrow (\mathrm{N1})_1] \\
 & [\mathrm{N3} \leftarrow (\mathrm{N1})_2] \\
 & [\mathrm{P1} \leftarrow *^{\leq}(\mathrm{N2})] \\
 & \left[\text{IF P1} \in \mathrm{Pro}^{\Sigma} \text{ GOTO L3}\right] \\
 & \text{GOTO L2} \\
\text{L3} & [\mathrm{N4} \leftarrow 10] \\
 & \left[\text{IF } Halt^{1,0}(\mathrm{N3},\mathrm{N4},\mathrm{P1}) \text{ GOTO L4}\right] \\
 & \text{GOTO L2} \\
\text{L4} & \left[\mathrm{N5} \leftarrow E^{1,0}_{\#1}(\mathrm{N3},\mathrm{N4},\mathrm{P1})\right] \\
 & [\text{IF N5} = 10 \text{ GOTO L4}] \\
\text{L2} & [\mathrm{P1} \leftarrow \mathrm{SKIP}] \\
\text{L4} & \text{SKIP}
\end{array}$$

Cuando $\Sigma \supseteq \Sigma_p$ podemos correr un programa $\mathcal{P} \in \mathrm{Pro}^{\Sigma}$ partiendo de un estado que asigne a sus variables alfabeticas programas (ya que los programas son meras palabras de Σ^*). En particular podriamos correr un programa $\mathcal{P}$ desde el estado $\|\mathcal{P}\|$. Llamaremos A al conjunto formado por aquellos programas $\mathcal{P}$ tales que $\mathcal{P}$ se detiene partiendo del estado $\|\mathcal{P}\|$. Es decir

$$A = \{\mathcal{P} \in \mathrm{Pro}^{\Sigma} : \exists t \in \omega \text{ tal que } Halt^{0,1}(t, \mathcal{P}, \mathcal{P}) = 1\}$$

Por ejemplo SKIP $\in A$. Dicho rapida y sugestivamente A es el conjunto formado por aquellos programas que se detienen partiendo de si mismos. Dejamos al lector hacer un programa que enumere a A. Como veremos mas adelante este conjunto, si bien es Σ-enumerable, no es Σ-computable.

3.4.3 Godel vence a Turing

Para probar que toda funcion Σ-Turing computable es Σ-recursiva debemos estudiar la recursividad del funcionamiento de las maquinas de Turing. Cabe destacar que tal como se lo explico en la Subseccion 3.1 supondremos siempre que el conjunto de estados de una maquina de Turing $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ es un alfabeto disjunto con Γ.

Primero probaremos algunos lemas basicos.

Lemma 108 *Sea $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ una maquina de Turing. Entonces*

(1) Des es un conjunto $(\Gamma \cup Q)$-p.r.

(2) $St : Des \to Q$ es una funcion $(\Gamma \cup Q)$-p.r.

Notese que la funcion δ de una maquina de Turing $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ no es $(\Gamma \cup Q)$-mixta. Sin envargo los siguientes predicados $(\Gamma \cup Q)$-mixtos contienen toda la informacion de δ:

$$\begin{array}{rcl} P_L : Q \times \Gamma \times Q \times \Gamma & \to & \omega \\ (p, \sigma, q, \gamma) & \to & \left\{ \begin{array}{ll} 1 & \text{si } \delta(q, \gamma) = (p, \sigma, L) \\ 0 & \text{caso contrario} \end{array} \right. \end{array}$$

$$\begin{array}{rcl} P_L : Q \times \Gamma \times Q \times \Gamma & \to & \omega \\ (p, \sigma, q, \gamma) & \to & \left\{ \begin{array}{ll} 1 & \text{si } (p, \sigma, L) \in \delta(q, \gamma) \\ 0 & \text{caso contrario} \end{array} \right. \end{array}$$

$$\begin{array}{rcl} P_R : Q \times \Gamma \times Q \times \Gamma & \to & \omega \\ (p, \sigma, q, \gamma) & \to & \left\{ \begin{array}{ll} 1 & \text{si } \delta(q, \gamma) = (p, \sigma, R) \\ 0 & \text{caso contrario} \end{array} \right. \end{array}$$

$$\begin{array}{rcl} P_K : Q \times \Gamma \times Q \times \Gamma & \to & \omega \\ (p, \sigma, q, \gamma) & \to & \left\{ \begin{array}{ll} 1 & \text{si } \delta(q, \gamma) = (p, \sigma, K) \\ 0 & \text{caso contrario} \end{array} \right. \end{array}$$

Lemma 109 *Sea $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ una maquina de Turing. Entonces los predicados P_L, P_R y P_K son $(\Gamma \cup Q)$-p.r.*

Proof. Ya que los tres predicados tienen dominio finito, el Corolario 59 nos dice que son $(\Gamma \cup Q)$-p.r. ■

Recordemos que dado $\alpha \in (Q \cup \Gamma)^*$, definimos $\lfloor \alpha \rfloor$ de la siguiente manera

$$\begin{array}{rcl} \lfloor \varepsilon \rfloor & = & \varepsilon \\ \lfloor \alpha\sigma \rfloor & = & \alpha\sigma, \text{ si } \sigma \neq B \\ \lfloor \alpha B \rfloor & = & \lfloor \alpha \rfloor \end{array}$$

Es decir $\lfloor \alpha \rfloor$ es el resultado de remover de α el tramo final mas grande de la forma B^n.

Tambien dada cualquier palabra α definimos

$$\begin{array}{rcl} {}^{\frown}\alpha & = & \left\{ \begin{array}{ll} [\alpha]_2 ... [\alpha]_{|\alpha|} & \text{si} \quad |\alpha| \geq 2 \\ \varepsilon & \text{si} \quad |\alpha| \leq 1 \end{array} \right. \\ \alpha^{\frown} & = & \left\{ \begin{array}{ll} [\alpha]_1 ... [\alpha]_{|\alpha|-1} & \text{si} \quad |\alpha| \geq 2 \\ \varepsilon & \text{si} \quad |\alpha| \leq 1 \end{array} \right. \end{array}$$

Lemma 110 *Las funciones $\lambda\alpha[\lfloor\alpha\rfloor]$, $\lambda\alpha[{}^{\frown}\alpha]$ y $\lambda\alpha[\alpha^{\frown}]$ son $(\Gamma \cup Q)$-p.r.. (Notar que la notacion λ aqui es respecto del alfabeto $\Gamma \cup Q$ por lo cual las tres funciones tienen dominio igual a $(\Gamma \cup Q)^*$.)*

Notese que dada una maquina de Turing M, la expresion $d \underset{M}{\vdash} d'$ fue definida solo en el caso en que d y d' son descripciones instantaneas. Es decir que el predicado $\lambda dd' [d \vdash d']$ tiene dominio igual a $Des \times Des$.

Lemma 111 *El predicado $\lambda dd' \left[d \underset{M}{\vdash} d' \right]$ es $(\Gamma \cup Q)$-p.r..*

Proof. Sea $\tilde{P}_L : Des \times Des \times \Gamma \times \Gamma^* \times \Gamma^* \times Q \times Q \to \omega$ definido por $\tilde{P}_L(d, d', \sigma, \alpha, \beta, p, q) = 1$ sii

$$d = \alpha p \beta \wedge (q, \sigma, L) = \delta\left(p, [\beta B]_1\right) \wedge \alpha \neq \varepsilon \wedge d' = \left\lfloor \alpha^{\frown} q \, [\alpha]_{|\alpha|} \, \sigma^{\frown} \beta \right\rfloor$$

Sea $\tilde{P}_R : Des \times Des \times \Gamma \times \Gamma^* \times \Gamma^* \times Q \times Q \to \omega$ definido por $\tilde{P}_R(d, d', \sigma, \alpha, \beta, p, q) = 1$ sii

$$d = \alpha p \beta \wedge (q, \sigma, R) = \delta\left(p, [\beta B]_1\right) \wedge d' = \alpha\sigma q^{\frown} \beta$$

Sea $\tilde{P}_K : Des \times Des \times \Gamma \times \Gamma^* \times \Gamma^* \times Q \times Q \to \omega$ definido por $\tilde{P}_K(d, d', \sigma, \alpha, \beta, p, q) = 1$ sii

$$d = \alpha p \beta \wedge (q, \sigma, K) = \delta\left(p, [\beta B]_1\right) \wedge d' = \lfloor \alpha q \sigma^{\frown} \beta \rfloor$$

Se deja al lector la verificacion de que estos predicados son $(\Gamma \cup Q)$-p.r.. Notese que $\lambda dd' \left[d \underset{M}{\vdash} d' \right]$ es igual al predicado

$$\lambda dd' \left[(\exists \sigma \in \Gamma)(\exists \alpha, \beta \in \Gamma^*)(\exists p, q \in Q)(\tilde{P}_R \vee \tilde{P}_L \vee \tilde{P}_K)(d, d', \sigma, \alpha, \beta, p, q) \right]$$

lo cual por el Lema 62 nos dice que $\lambda dd' \left[d \underset{M}{\vdash} d' \right]$ es $(\Gamma \cup Q)$-p.r. ■

Proposition 112 $\lambda ndd'\left[d \overset{n}{\underset{M}{\vdash}} d'\right]$ *es* $(\Gamma\cup Q)$*-p.r..*

Proof. Sea $Q=\lambda dd'\left[d \underset{M}{\vdash} d'\right]\cup C_0^{0,2}|_{(\Gamma\cup Q)^{*2}-Des^2}$ es decir Q es el resultado de extender con el valor 0 al predicado $\lambda dd'\left[d \underset{M}{\vdash} d'\right]$ de manera que este definido en todo $(\Gamma\cup Q)^{*2}$. Sea $\leq$ un orden total sobre $\Gamma\cup Q$ y sea $Q_1:\mathbf{N}\times Des\times Des\to\omega$ definido por $Q_1(x,d,d')=1$ sii

$$\big((\forall i\in\mathbf{N})_{i\leq Lt(x)} *^{\leq}((x)_i)\in Des\big)\wedge *^{\leq}((x)_1)=d\wedge$$
$$*^{\leq}((x)_{Lt(x)})=d'\wedge\Big((\forall i\in\mathbf{N})_{i\leq Lt(x)\dot{-}1}\ Q(*^{\leq}((x)_i),*^{\leq}((x)_{i+1}))\Big)$$

Notese que dicho rapidamente $Q_1(x,d,d')=1$ sii x codifica una computacion que parte de d y llega a d'. Se deja al lector la verificacion de que este predicado es $(\Gamma\cup Q)$-p.r.. Notese que

$$\lambda ndd'\left[d \overset{n}{\underset{M}{\vdash}} d'\right]=\lambda ndd'\left[(\exists x\in\mathbf{N})\ Lt(x)=n+1\wedge Q_1(x,d,d')\right]$$

Es decir que solo nos falta acotar el cuantificador existencial, para poder aplicar el lema de cuantificacion acotada. Ya que cuando $d_1,...,d_{n+1}\in Des$ son tales que $d_1\underset{M}{\vdash} d_2 \underset{M}{\vdash}\cdots\underset{M}{\vdash} d_{n+1}$ tenemos que

$$|d_i|\leq|d_1|+n,\text{ para }i=1,...,n$$

una posible cota para dicho cuantificador es

$$\prod_{i=1}^{n+1} pr(i)^{|\Gamma\cup Q|^{|d|+n}}.$$

O sea que, por el lema de cuantificacion acotada, tenemos que el predicado $\lambda ndd'\left[d \overset{n}{\underset{M}{\vdash}} d'\right]$ es $(\Gamma\cup Q)$-p.r. ■

@@finpagina@@

Theorem 113 *Supongamos* $f:S\subseteq\omega^n\times\Sigma^{*m}\to O$ *es* Σ*-Turing computable. Entonces* f *es* Σ*-recursiva.*

Proof. Supongamos $O=\Sigma^*$ y sea $M=(Q,\Sigma,\Gamma,\delta,q_0,B,\shortmid,F)$ una maquina de Turing deterministica con unit la cual compute a f. Sea $\leq$ un orden total sobre Σ. Notese que por el Teorema 84, la funcion $*^{\leq}$ es $(\Gamma\cup Q)$-p.r.. Sea $P:\mathbf{N}\times\omega^n\times\Sigma^{*m}\to\omega$ dado por $P(x,\vec{x},\vec{\alpha})=1$ sii

1. $(\exists q\in Q)\ \lfloor q_0B\shortmid^{x_1}...B\shortmid^{x_n}B\alpha_1...B\alpha_m\rfloor \overset{(x)_1}{\underset{M}{\vdash}} \lfloor qB *^{\leq}((x)_2)\rfloor\wedge$
2. $\wedge(\forall d\in Des)_{|d|\leq|*^{\leq}((x)_2)|+2}\neg\Big(\lfloor qB *^{\leq}((x)_2)\rfloor \underset{M}{\vdash} d\Big)$

Dejamos al lector la prueba de que P es $(\Gamma \cup Q)$-p.r.. Ya que P es Σ-mixto, el Teorema 84 nos dice que P es Σ-p.r.. Notese que

$$f = \lambda\vec{x}\vec{\alpha}\left[*^{\leq}\left(\left(\min_x P(x,\vec{x},\vec{\alpha})\right)_2\right)\right],$$

lo cual nos dice que f es Σ-recursiva. ■

3.4.4 Turing vence a Neumann

Probaremos que toda funcion Σ-computable es Σ-Turing computable. Para esto probaremos un resultado general que nos enseñara a simular el comportamiento de un programa con una maquina de Turing. Es importante notar que la simulacion que nos interesa que haga la maquina simuladora no es a nivel de la funcion que computa el programa sino a un nivel mas general, es decir nos interesa que simule a dicho programa como transformador de estados. En particular y usada adecuadamente, la maquina simuladora nos servira para confeccionar una maquina que compute una funcion computada por un programa dado.

Construccion de la maquina simuladora de un programa Dado $\mathcal{P} \in \mathrm{Pro}^{\Sigma}$, definamos

$$N(\mathcal{P}) \;=\; \text{menor } k \in \mathbf{N} \text{ tal que las variables que ocurren en } \mathcal{P} \text{ estan todas en la lista } \mathrm{N1},...,\mathrm{N}\bar{k},\mathrm{P1},...,\mathrm{P}\bar{k}$$

Por ejemplo si $\mathcal{P}$ es el siguiente programa (aqui $\Sigma = \{\blacktriangle, \#\}$)

$$\begin{array}{ll} \mathrm{L1} & \mathrm{N4} \leftarrow \mathrm{N4}+1 \\ & \mathrm{P1} \leftarrow \mathrm{P1}.\blacktriangle \\ & \mathrm{IF\ N1} \neq 0 \mathrm{\ GOTO\ L1} \end{array}$$

entonces tenemos $N(\mathcal{P}) = 4$

Sea $\mathcal{P}$ un programa y sea k fijo y mayor o igual a $N(\mathcal{P})$. La construccion de la maquina simuladora dependera de $\mathcal{P}$ y de k. Notese que cuando $\mathcal{P}$ se corre desde algun estado de la forma

$$\|x_1,...,x_k,\alpha_1,...,\alpha_k\|$$

los sucesivos estados por los que va pasando son todos de la forma

$$\|y_1,...,y_k,\beta_1,...,\beta_k\|$$

es decir en todos ellos las variables con indice mayor que k valen 0 o ε. La razon es simple: ya que en $\mathcal{P}$ no figuran las variables

$$\begin{array}{l} \mathrm{N}\overline{k+1},\mathrm{N}\overline{k+2},... \\ \mathrm{P}\overline{k+1},\mathrm{P}\overline{k+2},... \end{array}$$

estas variables quedan con valores 0 y ε, respectivamente a lo largo de toda la computacion.

La maquina simuladora que construiremos simulara a $\mathcal{P}$ en cuanto a su funcionamiento cuando partimos de estados de la forma $\|x_1,...,x_k,\alpha_1,...,\alpha_k\|$. Necesitaremos tener alguna manera de representar en la cinta los diferentes estados por los cuales se va pasando, a medida que corremos a $\mathcal{P}$. Esto lo haremos de la siguiente forma: al estado

$$\|x_1,...,x_k,\alpha_1,...,\alpha_k\|$$

lo representaremos en la cinta de la siguiente manera

$$B \shortmid^{x_1} ...B \shortmid^{x_k} B\alpha_1...B\alpha_k BBBB....$$

Por ejemplo consideremos el programa $\mathcal{P}$ mostrado recien y fijemos $k = 6$. Entonces al estado

$$\|3,2,5,0,4,2,\blacktriangle,\blacktriangle\blacktriangle,\varepsilon,\#\blacktriangle,\#,\#\#\#\|$$

lo representaremos en la cinta de la siguiente manera

$$B{\shortmid}{\shortmid}{\shortmid}B{\shortmid}{\shortmid}B{\shortmid}{\shortmid}{\shortmid}{\shortmid}{\shortmid}BB{\shortmid}{\shortmid}{\shortmid}{\shortmid}B{\shortmid}{\shortmid}B\blacktriangle B\blacktriangle\blacktriangle BB\#\blacktriangle B\#B\#\#\#BBBBB....$$

A lo que queda entre dos blancos consecutivos (es decir que no hay ningun blanco entre ellos) lo llamaremos "bloque", por ejemplo en la cinta de arriba tenemos que los primeros 12 bloques son

$${\shortmid}{\shortmid}{\shortmid} \quad {\shortmid}{\shortmid} \quad {\shortmid}{\shortmid}{\shortmid}{\shortmid}{\shortmid} \quad \varepsilon \quad {\shortmid}{\shortmid}{\shortmid}{\shortmid} \quad {\shortmid}{\shortmid} \quad \blacktriangle \quad \blacktriangle\blacktriangle \quad \varepsilon \quad \#\blacktriangle \quad \# \quad \#\#\#$$

y despues los bloques siguientes (que son infinitos ya que la cinta es infinita hacia la derecha) son todos iguales a ε.

Una observacion importante es que esta forma de representacion de estados en la cinta depende del k elejido, es decir si tomaramos otro k, por ejemplo $k = 9$, entonces el estado anterior se representaria de otra forma en la cinta. Aqui se ve claramente que la maquina simuladora que construiremos dependera del k elejido.

Construccion de las maquinas simuladoras de instrucciones Armaremos la maquina simuladora como concatenacion de maquinas las cuales simularan, via la representacion anterior, el funcionamiento de las distintas instrucciones de $\mathcal{P}$. Asumiremos que en $\mathcal{P}$ no hay instrucciones de la forma GOTO L$\bar{m}$ ni de la forma L$\bar{n}$ GOTO L$\bar{m}$. Esto simplificara un poco la construccion de la maquina simuladora y de hecho lo podemos hacer ya que toda funcion Σ-computable puede ser computada por un programa sin este tipo de instrucciones, tal como lo veremos mas adelante (Lema 114).

Para poder hacer concretamente las maquinas simuladoras de instrucciones deberemos diseñar antes algunas maquinas auxiliares. Todas las maquinas descriptas a continuacion tendran a $\shortmid$ como unit y a B como blanco, tendran a Σ

como su alfabeto terminal y su alfabeto mayor sera $\Gamma = \Sigma \cup \{B, \shortmid\} \cup \{\tilde{a} : a \in \Sigma \cup \{\shortmid\}\}$. Ademas tendran uno o dos estados finales con la propiedad de que si q es un estado final, entonces $(q, \sigma) \notin D_\delta$, para cada $\sigma \in \Gamma$.

Para cada $j \geq 1$, sea D_j la siguiente maquina:

@@figura:figure1.png@@

Notese que

$$\begin{array}{lcr} \alpha B\beta_1 B\beta_2 B...B\beta_j B\gamma & \overset{*}{\vdash} & \alpha B\beta_1 B\beta_2 B...B\beta_j B\gamma \\ \uparrow & & \uparrow\; \\ q_0 & & q_f\; \end{array}$$

siempre que $\alpha, \gamma \in \Gamma^*$, $\beta_1, ..., \beta_j \in (\Gamma - \{B\})^*$. Es decir la maquina D_j lo unico que hace es mover el cabezal desde el blanco de la izquierda de un bloque determinado, exactamente j bloques a la derecha

Analogamente I_j sera una maquina que desplaza el cabezal j bloques a la izquierda del blanco que esta escaneando. Es decir I_j cumplira que

$$\begin{array}{rcl} \alpha B\beta_j B...B\beta_2 B\beta_1 B\gamma & \overset{*}{\vdash} & \alpha B\beta_j B...B\beta_2 B\beta_1 B\gamma \\ \uparrow\;\; & & \uparrow \\ q_0\;\; & & q_f \end{array}$$

siempre que $\alpha, \gamma \in \Gamma^*$, $\beta_1, ..., \beta_j \in (\Gamma - \{B\})^*$. Dejamos al lector la manufactura de esta maquina.

Para $j \geq 1$, sea TD_j una maquina con un solo estado final q_f y tal que

$$\begin{array}{lcl} \alpha B\gamma & \overset{*}{\vdash} & \alpha BB\gamma \\ \;\uparrow & & \;\;\uparrow \\ \;q_0 & & \;\;q_f \end{array}$$

cada vez que $\alpha, \gamma \in \Gamma^*$ y γ tiene exactamente j ocurrencias de B. Es decir la maquina TD_j corre un espacio a la derecha todo el segmento γ y agrega un blanco en el espacio que se genera a la izquierda. Por ejemplo, para el caso de $\Sigma = \{a\}$ podemos tomar TD_3 igual a la siguiente maquina:

@@figura:figure2.png@@

Analogamente, para $j \geq 1$, sea TI_j una maquina tal que

$$\begin{array}{lcl} \alpha B\sigma\gamma & \overset{*}{\vdash} & \alpha B\gamma \\ \;\uparrow & & \;\;\uparrow \\ \;q_0 & & \;\;q_f \end{array}$$

cada vez que $\alpha \in \Gamma^*$, $\sigma \in \Gamma$ y γ tiene exactamente j ocurrencias de B. Es decir la maquina TI_j corre un espacio a la izquierda todo el segmaneto γ (por lo cual en el lugar de σ queda el primer simbolo de γ). Dejamos al lector la construccion de por ejemplo TI_3 para $\Sigma = \{a\}$.

A continuacion describiremos las distintas maquinas simuladoras de instrucciones (y para algunos casos mostraremos concretamente como pueden ser hechas usando las maquinas anteriores).

Para $1 \leq i \leq k$, sea $M_{i,k}^{+}$ una maquina tal que cualesquiera sean $x_1, ..., x_k \in \omega$ y $\alpha_1, ..., \alpha_k \in \Sigma^*$:

$$\begin{array}{lcl} B \shortmid^{x_1} ...B \shortmid^{x_k} B\alpha_1...B\alpha_k & \overset{*}{\vdash} & B \shortmid^{x_1} ...B \shortmid^{x_{i-1}} B \shortmid^{x_i+1} B \shortmid^{x_{i+1}} ...B \shortmid^{x_k} B\alpha_1...B\alpha_k \\ \uparrow & & \uparrow \\ q_0 & & q_f \end{array}$$

donde q_0 es el estado inicial y q_f es el unico estado final de $M_{i,k}^{+}$. Es claro que la maquina $M_{i,k}^{+}$ simula la instruccion N$\bar{\imath} \leftarrow$ N$\bar{\imath}+1$, via la representacion de estados en la cinta con respecto a k.

Para $1 \leq i \leq k$, sea $M_{i,k}^{\dot{-}}$ una maquina tal que cualesquiera sean $x_1, ..., x_k \in \omega$ y $\alpha_1, ..., \alpha_k \in \Sigma^*$:

$$\begin{array}{lcl} B \shortmid^{x_1} ...B \shortmid^{x_k} B\alpha_1...B\alpha_k & \overset{*}{\vdash} & B \shortmid^{x_1} ...B \shortmid^{x_{i-1}} B \shortmid^{x_i \dot{-} 1} B \shortmid^{x_{i+1}} ...B \shortmid^{x_k} B\alpha_1...B\alpha_k \\ \uparrow & & \uparrow \\ q_0 & & q_f \end{array}$$

donde q_0 es el estado inicial y q_f es el unico estado final de $M_{i,k}^{\dot{-}}$. Es claro que la maquina $M_{i,k}^{\dot{-}}$ simula la instruccion P$\bar{\imath} \leftarrow$ P$\bar{\imath}\dot{-}1$, via la representacion de estados en la cinta con respecto a k.

Para $1 \leq i \leq k$ y $a \in \Sigma$, sea $M_{i,k}^{a}$ una maquina tal que cualesquiera sean $x_1, ..., x_k \in \omega$ y $\alpha_1, ..., \alpha_k \in \Sigma^*$:

$$\begin{array}{lcl} B \shortmid^{x_1} ...B \shortmid^{x_k} B\alpha_1...B\alpha_k & \overset{*}{\vdash} & B \shortmid^{x_1} ...B \shortmid^{x_k} B\alpha_1...B\alpha_{i-1}B\alpha_i aB\alpha_{i+1}...B\alpha_k \\ \uparrow & & \uparrow \\ q_0 & & q_f \end{array}$$

donde q_0 es el estado inicial y q_f es el unico estado final de $M_{i,k}^{a}$. Es claro que la maquina $M_{i,k}^{a}$ simula la instruccion P$\bar{\imath} \leftarrow$ P$\bar{\imath}.a$, via la representacion de estados en la cinta con respecto a k. La maquina $M_{i,k}^{a}$.puede hacerse de la siguiente manera:

@@figura:figure3.png@@

Para $1 \leq i \leq k$, sea $M_{i,k}^{\curvearrowright}$ una maquina tal que cualesquiera sean $x_1, ..., x_k \in \omega$ y $\alpha_1, ..., \alpha_k \in \Sigma^*$:

$$\begin{array}{lcl} B \shortmid^{x_1} ...B \shortmid^{x_k} B\alpha_1...B\alpha_k & \overset{*}{\vdash} & B \shortmid^{x_1} ...B \shortmid^{x_k} B\alpha_1...B\alpha_{i-1}B^{\curvearrowright}\alpha_i B\alpha_{i+1}...B\alpha_k \\ \uparrow & & \uparrow \\ q_0 & & q_f \end{array}$$

donde q_0 es el estado inicial y q_f es el unico estado final de $M_{i,k}^{\curvearrowright}$. Es claro que la maquina $M_{i,k}^{\curvearrowright}$ simula la instruccion P$\bar{\imath} \leftarrow {}^{\curvearrowright}P\bar{\imath}$, via la representacion de estados en la cinta con respecto a k.

Para $1 \leq i, j \leq k$, sea $M_{i \leftarrow j}^{\#,k}$ una maquina tal que cualesquiera sean $x_1, ..., x_k \in \omega$ y $\alpha_1, ..., \alpha_k \in \Sigma^*$:

$$\begin{array}{lcl} B \shortmid^{x_1} ...B \shortmid^{x_k} B\alpha_1...B\alpha_k & \overset{*}{\vdash} & B \shortmid^{x_1} ...B \shortmid^{x_{i-1}} B \shortmid^{x_j} B \shortmid^{x_{i+1}} ...B \shortmid^{x_k} B\alpha_1...B\alpha_k \\ \uparrow & & \uparrow \\ q_0 & & q_f \end{array}$$

donde q_0 es el estado inicial y q_f es el unico estado final de $M_{i\leftarrow j}^{\#,k}$. Es claro que la maquina $M_{i\leftarrow j}^{\#,k}$ simula la instruccion N$\bar{\imath} \leftarrow$ N$\bar{j}$, via la representacion de estados en la cinta con respecto a k.

Para $1 \leq i,j \leq k$, sea $M_{i\leftarrow j}^{*,k}$ una maquina tal que cualesquiera sean $x_1,...,x_k \in \omega$ y $\alpha_1,...,\alpha_k \in \Sigma^*$:

$$\begin{array}{lcl} B\mid^{x_1}...B\mid^{x_k}B\alpha_1...B\alpha_k & \overset{*}{\vdash} & B\mid^{x_1}...B\mid^{x_k}B\alpha_1...B\alpha_{i-1}B\alpha_jB\alpha_{i+1}...B\alpha_k \\ \uparrow & & \uparrow \\ q_0 & & q_f \end{array}$$

donde q_0 es el estado inicial y q_f es el unico estado final de $M_{i\leftarrow j}^{*,k}$. Es claro que la maquina $M_{i\leftarrow j}^{*,k}$ simula la instruccion P$\bar{\imath} \leftarrow$ P$\bar{j}$, via la representacion de estados en la cinta con respecto a k. La maquina $M_{i\leftarrow j}^{*,k}$, para el caso $\Sigma = \{a,b\}$ y $i < j$ puede hacerse de la siguiente manera:

@@figura:figure4.png@@

Para $1 \leq i \leq k$, sea $M_{i\leftarrow 0}^{k}$ una maquina tal que cualesquiera sean $x_1,...,x_k \in \omega$ y $\alpha_1,...,\alpha_k \in \Sigma^*$:

$$\begin{array}{lcl} B\mid^{x_1}...B\mid^{x_k}B\alpha_1...B\alpha_k & \overset{*}{\vdash} & B\mid^{x_1}...B\mid^{x_{i-1}}BB\mid^{x_{i+1}}...B\mid^{x_k}B\alpha_1...B\alpha_k \\ \uparrow & & \uparrow \\ q_0 & & q_f \end{array}$$

donde q_0 es el estado inicial y q_f es el unico estado final de $M_{i\leftarrow 0}^{k}$. Es claro que la maquina $M_{i\leftarrow 0}^{k}$ simula la instruccion N$\bar{\imath} \leftarrow 0$, via la representacion de estados en la cinta con respecto a k.

Para $1 \leq i \leq k$, sea $M_{i\leftarrow \varepsilon}^{k}$ una maquina tal que cualesquiera sean $x_1,...,x_k \in \omega$ y $\alpha_1,...,\alpha_k \in \Sigma^*$:

$$\begin{array}{lcl} B\mid^{x_1}...B\mid^{x_k}B\alpha_1...B\alpha_k & \overset{*}{\vdash} & B\mid^{x_1}...B\mid^{x_k}B\alpha_1...B\alpha_{i-1}BB\alpha_{i+1}...B\alpha_k \\ \uparrow & & \uparrow \\ q_0 & & q_f \end{array}$$

donde q_0 es el estado inicial y q_f es el unico estado final de $M_{i\leftarrow \varepsilon}^{k}$. Es claro que la maquina $M_{i\leftarrow \varepsilon}^{k}$ simula la instruccion P$\bar{\imath} \leftarrow \varepsilon$, via la representacion de estados en la cinta con respecto a k.

Sea

$$M_{\text{SKIP}} = (\{q_0,q_f\},\Gamma,\Sigma,\delta,q_0,B,\mid,\{q_f\}),$$

con $D_\delta = \{(q_0,B)\}$ y $\delta(q_0,B) = (q_f,B,K)$. Es claro que la maquina M_{SKIP} simula la instruccion SKIP, via la representacion de estados en la cinta con respecto a k (cualquiera sea el k).

Para $1 \leq j \leq k$, sea $IF_{j,k}$ una maquina con estado inicial q_0 y dos estados finales q_{si} y q_{no} tal que cualesquiera sean $x_1,...,x_k \in \omega$ y $\alpha_1,...,\alpha_k \in \Sigma^*$, si $x_j \neq 0$, entonces

$$\begin{array}{lcl} B\mid^{x_1}...B\mid^{x_k}B\alpha_1...B\alpha_k & \overset{*}{\vdash} & B\mid^{x_1}...B\mid^{x_k}B\alpha_1...B\alpha_k \\ \uparrow & & \uparrow \\ q_0 & & q_{si} \end{array}$$

y si $x_j = 0$, entonces

$$\begin{array}{lcl} B\shortmid^{x_1}...B\shortmid^{x_k}B\alpha_1...B\alpha_k & \overset{*}{\vdash} & B\shortmid^{x_1}...B\shortmid^{x_k}B\alpha_1...B\alpha_k \\ \uparrow & & \uparrow \\ q_0 & & q_{no} \end{array}$$

Para $1 \leq i \leq k$ y $a \in \Sigma$, sea $IF_{j,k}^{a}$ una maquina con estado inicial q_0 y dos estados finales q_{si} y q_{no} tal que cualesquiera sean $x_1, ..., x_k \in \omega$ y $\alpha_1, ..., \alpha_k \in \Sigma^*$, si α_j comienza con a, entonces

$$\begin{array}{lcl} B\shortmid^{x_1}...B\shortmid^{x_k}B\alpha_1...B\alpha_k & \overset{*}{\vdash} & B\shortmid^{x_1}...B\shortmid^{x_k}B\alpha_1...B\alpha_k \\ \uparrow & & \uparrow \\ q_0 & & q_{si} \end{array}$$

y en caso contrario

$$\begin{array}{lcl} B\shortmid^{x_1}...B\shortmid^{x_k}B\alpha_1...B\alpha_k & \overset{*}{\vdash} & B\shortmid^{x_1}...B\shortmid^{x_k}B\alpha_1...B\alpha_k \\ \uparrow & & \uparrow \\ q_0 & & q_{no} \end{array}$$

La maquina $IF_{j,k}^{a}$ puede hacerse de la siguinete manera:
@@figura:figure5.png@@

Ejemplo de maquina simuladora de un programa A continuacion veremos un ejemplo de como se arma la maquina simuladora de un programa dado. Sea $\Sigma = \{\blacktriangle, \#\}$ y sea $\mathcal{P}$ el siguiente programa

$$\begin{array}{ll} \mathrm{L3} & \mathrm{N4} \leftarrow \mathrm{N4}+1 \\ & \mathrm{P1} \leftarrow {}^{\curvearrowright}\mathrm{P1} \\ & \mathrm{IF\ P1\ BEGINS} \blacktriangle \mathrm{\ GOTO\ L3} \\ & \mathrm{P3} \leftarrow \mathrm{P3}.\# \end{array}$$

Tomemos $k = 5$. Es claro que $k \geq N(\mathcal{P}) = 4$. A la maquina que simulara a $\mathcal{P}$ respecto de k, la llamaremos M_{sim} y sera la siguiente maquina:
@@figura:figure6.png@@
Veamos con un ejemplo como M_{sim} simula a $\mathcal{P}$. Supongamos que corremos $\mathcal{P}$ desde el estado

$$\|2, 1, 0, 5, 3, \#\blacktriangle\#\#, \varepsilon, \blacktriangle\blacktriangle, \#\blacktriangle, \#\|$$

Tendremos entonces la siguiente sucesion de descripciones instantaneas:

$$(1, \|2, 1, 0, 5, 3, \#\blacktriangle\#\#, \varepsilon, \blacktriangle\blacktriangle, \#\blacktriangle, \#\|)$$

$$(2, \|2, 1, 0, 6, 3, \#\blacktriangle\#\#, \varepsilon, \blacktriangle\blacktriangle, \#\blacktriangle, \#\|)$$

$$(3, \|2, 1, 0, 6, 3, \blacktriangle\#\#, \varepsilon, \blacktriangle\blacktriangle, \#\blacktriangle, \#\|)$$

$$(1, \|2, 1, 0, 6, 3, \blacktriangle\#\#, \varepsilon, \blacktriangle\blacktriangle, \#\blacktriangle, \#\|)$$

$$(2, \|2, 1, 0, 7, 3, \blacktriangle\#\#, \varepsilon, \blacktriangle\blacktriangle, \#\blacktriangle, \#\|)$$

$$(3, \|2, 1, 0, 7, 3, \#\#, \varepsilon, \blacktriangle\blacktriangle, \#\blacktriangle, \#\|)$$

$$(4, \|2, 1, 0, 7, 3, \#\#, \varepsilon, \blacktriangle\blacktriangle, \#\blacktriangle, \#\|)$$

$$(5, \|2, 1, 0, 7, 3, \#\#, \varepsilon, \blacktriangle\blacktriangle\#, \#\blacktriangle, \#\|)$$

Si hacemos funcionar a M_{sim} desde $q_0B \shortmid^2 B \shortmid BB \shortmid^5 B \shortmid^3 B\#\blacktriangle\#\#BB\blacktriangle\blacktriangle B\#\blacktriangle B\#B$ obtendremos una sucesion de descripciones instantaneas dentro de la cual estara la siguiente subsucesion que se corresponde con las descripciones instantaneas de la computacion anterior.

$$q_0B \quad \shortmid \quad {}^2B \shortmid BB \shortmid^5 B \shortmid^3 B\#\blacktriangle\#\#BB\blacktriangle\blacktriangle B\#\blacktriangle B\#B$$

$$q_1B \quad \shortmid \quad {}^2B \shortmid BB \shortmid^6 B \shortmid^3 B\#\blacktriangle\#\#BB\blacktriangle\blacktriangle B\#\blacktriangle B\#B$$
$$q_2B \quad \shortmid \quad {}^2B \shortmid BB \shortmid^6 B \shortmid^3 B\#\blacktriangle\#\#BB\blacktriangle\blacktriangle B\#\blacktriangle B\#B$$

$$q_3B \quad \shortmid \quad {}^2B \shortmid BB \shortmid^6 B \shortmid^3 B\blacktriangle\#\#BB\blacktriangle\blacktriangle B\#\blacktriangle B\#B$$
$$q_4B \quad \shortmid \quad {}^2B \shortmid BB \shortmid^6 B \shortmid^3 B\blacktriangle\#\#BB\blacktriangle\blacktriangle B\#\blacktriangle B\#B$$

$$q_{si}B \quad \shortmid \quad {}^2B \shortmid BB \shortmid^6 B \shortmid^3 B\blacktriangle\#\#BB\blacktriangle\blacktriangle B\#\blacktriangle B\#B$$
$$q_0B \quad \shortmid \quad {}^2B \shortmid BB \shortmid^6 B \shortmid^3 B\blacktriangle\#\#BB\blacktriangle\blacktriangle B\#\blacktriangle B\#B$$

$$q_1B \quad \shortmid \quad {}^2B \shortmid BB \shortmid^7 B \shortmid^3 B\blacktriangle\#\#BB\blacktriangle\blacktriangle B\#\blacktriangle B\#B$$
$$q_2B \quad \shortmid \quad {}^2B \shortmid BB \shortmid^7 B \shortmid^3 B\blacktriangle\#\#BB\blacktriangle\blacktriangle B\#\blacktriangle B\#B$$

$$q_3B \quad \shortmid \quad {}^2B \shortmid BB \shortmid^7 B \shortmid^3 B\#\#BB\blacktriangle\blacktriangle B\#\blacktriangle B\#B$$
$$q_4B \quad \shortmid \quad {}^2B \shortmid BB \shortmid^7 B \shortmid^3 B\#\#BB\blacktriangle\blacktriangle B\#\blacktriangle B\#B$$

$$q_{no}B \quad \shortmid \quad {}^2B \shortmid BB \shortmid^7 B \shortmid^3 B\#\#BB\blacktriangle\blacktriangle B\#\blacktriangle B\#B$$
$$q_5B \quad \shortmid \quad {}^2B \shortmid BB \shortmid^7 B \shortmid^3 B\#\#BB\blacktriangle\blacktriangle B\#\blacktriangle B\#B$$

$$q_6B \quad \shortmid \quad {}^2B \shortmid BB \shortmid^7 B \shortmid^3 B\#\#BB\blacktriangle\blacktriangle\#B\#\blacktriangle B\#B$$

Dejamos al lector ver en detalle el paralelismo que hay entre las dos sucesiones de descripciones instantaneas arriba expuestas.

La contruccion de la maquina simuladora A continuacion describiremos en general como hacer la maquina simuladora de $\mathcal{P}$, respecto de k. Supongamos que $\mathcal{P} = I_1...I_n$. Para cada $i = 1,...,n$, llamaremos M_i a la maquina que simulara el efecto que produce la instruccion I_i, es decir tomemos

- $M_i = M^{+}_{j,k}$, si $Bas(I_i) = \mathrm{N}\bar{j} \leftarrow \mathrm{N}\bar{j} + 1$
- $M_i = M^{\dot{-}}_{j,k}$, si $Bas(I_i) = \mathrm{N}\bar{j} \leftarrow \mathrm{N}\bar{j} \dot{-} 1$
- $M_i = M^{a}_{j,k}$, si $Bas(I_i) = \mathrm{P}\bar{j} \leftarrow \mathrm{P}\bar{j}.a$
- $M_i = M^{\frown}_{j,k}$, si $Bas(I_i) = \mathrm{P}\bar{j} \leftarrow {}^{\frown}\mathrm{P}\bar{j}$
- $M_i = M^{\#,k}_{j\leftarrow m}$, si $Bas(I_i) = \mathrm{N}\bar{j} \leftarrow \mathrm{N}\bar{m}$

- $M_i = M^{*,k}_{j\leftarrow m}$, si $Bas(I_i) = \text{P}\bar{j} \leftarrow \text{P}\bar{m}$

- $M_i = M^{k}_{j\leftarrow 0}$, si $Bas(I_i) = \text{N}\bar{j} \leftarrow 0$

- $M_i = M^{k}_{j\leftarrow \varepsilon}$, si $Bas(I_i) = \text{P}\bar{j} \leftarrow \varepsilon$

- $M_i = M_{\text{SKIP}}$, si $Bas(I_i) = \text{SKIP}$

- $M_i = IF_{j,k}$, si $Bas(I_i) = \text{IF N}\bar{j} \neq 0 \text{ GOTO L}\bar{m}$, para algun m

- $M_i = IF^{a}_{j,k}$, si $Bas(I_i) = \text{IF P}\bar{j} \text{ BEGINS } a \text{ GOTO L}\bar{m}$, para algun m

Ya que la maquina M_i puede tener uno o dos estados finales, la representaremos como se muestra a continuacion:
@@figura:figure7.png@@
entendiendo que en el caso en que M_i tiene un solo estado final, este esta representado por el circulo de abajo a la izquierda y en el caso en que M_i tiene dos estados finales, el circulo de abajo a la izquierda corresponde al estado final q_{no} y el circulo de abajo a la derecha corresponde al estado q_{si}. Para armar la maquina que simulara a $\mathcal{P}$ hacemos lo siguiente. Primero unimos las maquinas $M_1, ..., M_n$ de la siguiente manera:
@@figura:figure8.png@@
Luego para cada i tal que $Bas(I_i)$ es de la forma αGOTO L$\bar{m}$, ligamos con una flecha de la forma

$$\xrightarrow{\quad B,B,K \quad}$$

el estado final q_{si} de la M_i con el estado inicial de la M_h, donde h es tal que I_h es la primer instruccion que tiene label L$\bar{m}$.

El lema de la simulacion A continuacion enunciaremos en forma de lema la existencia de la maquina simuladora y de las propiedades esenciales que usaremos luego para probar que toda funcion Σ-computable es Σ-Turing computable.

Lemma 114 *Sea $\mathcal{P} \in \text{Pro}^{\Sigma}$ y sea $k \geq N(\mathcal{P})$. Supongamos que en $\mathcal{P}$ no hay instrucciones de la forma* GOTO L$\bar{m}$ *ni de la forma* L$\bar{n}$ GOTO L$\bar{m}$. *Para cada $a \in \Sigma \cup \{\shortmid\}$, sea $\tilde{a}$ un nuevo simbolo. Sea $\Gamma = \Sigma \cup \{B, \shortmid\} \cup \{\tilde{a} : a \in \Sigma \cup \{\shortmid\}\}$. Entonces hay una maquina de Turing deterministica con unit $M = (Q, \Gamma, \Sigma, \delta, q_0, B, \shortmid, \{q_f\})$ la cual satisface*

(1) $(q_f, \sigma) \notin D_\delta$, para cada $\sigma \in \Gamma$.

(2) Cualesquiera sean $x_1, ..., x_k \in \omega$ y $\alpha_1, ..., \alpha_k \in \Sigma^*$, el programa $\mathcal{P}$ se detiene partiendo del estado

$$\|x_1, ..., x_k, \alpha_1, ..., \alpha_k\|$$

sii M se detiene partiendo de la descripcion instantanea

$$\lfloor q_0 B \shortmid^{x_1} B ... B \shortmid^{x_k} B\alpha_1 B ... B\alpha_k B \rfloor$$

(3) Si $x_1, ..., x_k \in \omega$ y $\alpha_1, ..., \alpha_k \in \Sigma^*$ son tales que $\mathcal{P}$ se detiene partiendo del estado

$$\|x_1, ..., x_k, \alpha_1, ..., \alpha_k\|$$

y llega al estado

$$\|y_1, ..., y_k, \beta_1, ..., \beta_k\|$$

entonces

$$\lfloor q_0 B \mid^{x_1} B ... B \mid^{x_k} B\alpha_1 B ... B\alpha_k B \rfloor \overset{*}{\underset{M}{\vdash}} \lfloor q_f B \mid^{y_1} B ... B \mid^{y_k} B\beta_1 B ... B\beta_k B \rfloor$$

Cabe destacar que si bien la veracidad de este lema es sustentada en las explicaciones anteriores, una prueba formal rigurosa del mismo resultaria extremadamente larga y tediosa. La ventaja de que sea un resultado intuitivamente claro nos permite aceptarlo y seguir adelante en nuestro analisis.

Turing vence a Neumann En lo que sigue usaremos la existencia de la maquina simuladora de un programa para probar que toda funcion Σ-computable es Σ-Turing computable. Antes un lema.

Lemma 115 *Si $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to \Sigma^*$ es Σ-computable, entonces hay un programa $\mathcal{Q}$ el cual computa a f y el cual cumple con las siguientes propiedades*

(1) En $\mathcal{Q}$ no hay instrucciones de la forma GOTO L$\bar{\imath}$ *ni de la forma* L$\bar{j}$ GOTO L$\bar{\imath}$

(2) Cuando $\mathcal{Q}$ termina partiendo de un estado cualquiera dado, el estado alcansado es tal que las variables numericas tienen todas el valor 0 *y las alfabeticas tienen todas exepto* P1 *el valor ε.*

Proof. Sea $\mathcal{P}$ un programa que compute a f. Sea $r \in \mathbf{N}$ tal que $r > N(\mathcal{P}), n, m$. Sea $\tilde{\mathcal{P}}$ el resultado de reemplazar en $\mathcal{P}$ cada instruccion de la forma

$$\alpha\text{GOTO L}\bar{\imath}$$

con $\alpha \in \{\varepsilon\} \cup \{\text{L}\bar{j} : j \in \mathbf{N}\}$ por αIF N$\bar{r} \neq 0$ GOTO L$\bar{\imath}$. Ahora sea $\mathcal{Q}$ el siguiente programa

$$\begin{array}{l} \text{N}\bar{r} \leftarrow \text{N}\bar{r} + 1 \\ \tilde{\mathcal{P}} \\ \text{N1} \leftarrow 0 \\ \vdots \\ \text{N}\bar{r} \leftarrow 0 \\ \text{P2} \leftarrow \varepsilon \\ \vdots \\ \text{P}\bar{r} \leftarrow \varepsilon \end{array}$$

Es facil ver que $\mathcal{Q}$ tiene las propiedades (1) y (2). ■

Por supuesto, hay un lema analogo para el caso en que f llega a ω en lugar de llegar a Σ^*. Ahora si, el anunciado teorema:

Theorem 116 *Si* $f : D_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$ *es* Σ*-computable, entonces* f *es* Σ*-Turing computable.*

Proof. Supongamos $O = \Sigma^*$. Por el Lema 115 existe $\mathcal{P} \in \mathrm{Pro}^{\Sigma}$ el cual computa f y tiene las propiedades (1) y (2). Sea $k = \max\{n, m, N(\mathcal{P})\}$ y sea M_{sim} la maquina de Turing con unit que simula a $\mathcal{P}$ respecto de k. Como puede observarse, la maquina M_{sim}, no necesariamente computara a f. Sea M_1 la siguiente maquina:

@@figura:figure9.png@@

(Cuando $n = 0$ debemos interpretar que $D_0 = (\{q_0, q_f\}, \Gamma, \Sigma, \delta, q_0, B, \shortmid, \{q_f\})$, con $D_\delta = \{(q_0, B)\}$ y $\delta(q_0, B) = (q_f, B, K)$. Notese que M_1 cumple que para cada $(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m}$,

$$\left\lfloor q_0 B \shortmid^{x_1} B...B \shortmid^{x_n} B\alpha_1 B...B\alpha_m B \right\rfloor \overset{*}{\vdash} \left\lfloor q_f B \shortmid^{x_1} B...B \shortmid^{x_n} B^{k-n} B\alpha_1 B...B\alpha_m B \right\rfloor$$

Sea M_2 la siguiente maquina

@@figura:figure10.png@@

Notese que M_2 cumple que para cada $\alpha \in \Sigma^*$,

$$\left\lfloor q_0 B^{k+1}\alpha \right\rfloor \overset{*}{\vdash} \lfloor q_f B\alpha \rfloor$$

Sea M la siguiente maquina:

@@figura:figure11.png@@

A continuacion veremos que M computa a f. Supongamos que $(\vec{x}, \vec{\alpha}) \in (\omega^n \times \Sigma^{*m}) - D_f$. Deberemos ver que M no termina partiendo de

(*) $\lfloor q_0 B \shortmid^{x_1} B...B \shortmid^{x_n} B\alpha_1 B...B\alpha_m B \rfloor$

Primero notemos que, ya que $\mathcal{P}$ computa a f, tenemos que $\mathcal{P}$ no termina partiendo de $\|x_1, ..., x_n, \alpha_1, ..., \alpha_m\|$ por lo cual $\mathcal{P}$ no termina partiendo de

$$\left\| x_1, ..., x_n, \overbrace{0, ..., 0}^{k-n}, \alpha_1, ..., \alpha_m, \overbrace{\varepsilon, ..., \varepsilon}^{k-m} \right\|$$

lo cual implica (Lema 114) que

(**) M_{sim} no termina partiendo de $\left\lfloor q_0 B \shortmid^{x_1} B...B \shortmid^{x_n} B^{k-n} B\alpha_1 B...B\alpha_m B \right\rfloor$

Ahora notese que si hacemos funcionar a M desde la descripcion instantanea dada en (*), llegaremos (via la copia de M_1 dentro de M) indefectiblemente (ya que M es deterministica) a la siguiente descripcion instantanea

$$\left\lfloor q_2 B \shortmid^{x_1} B...B \shortmid^{x_n} B^{k-n} B\alpha_1 B...B\alpha_m B \right\rfloor$$

Luego entonces (**) nos dice que al seguir trabajando M (ahora via la copia de M_{sim} dentro de M), la maquina M nunca terminara.

Para terminar de ver que M computa a f, tomemos $(\vec{x},\vec{\alpha}) \in D_f$ y veamos que

$$\lfloor q_0 B \shortmid^{x_1} B...B \shortmid^{x_n} B\alpha_1 B...B\alpha_m B \rfloor \overset{*}{\underset{M}{\vdash}} \lfloor q_5 B f(\vec{x},\vec{\alpha}) \rfloor$$

y que la maquina M se detiene en $\lfloor q_5 B f(\vec{x},\vec{\alpha}) \rfloor$. La maquina M se detiene en $\lfloor q_5 B f(\vec{x},\vec{\alpha}) \rfloor$ ya que q_5 es el estado final de una copia de M_2 y por lo tanto no sale ninguna flecha desde el. Ya que $\mathcal{P}$ computa a f y tiene la propiedad (2) del Lema 115, tenemos que $\mathcal{P}$ termina partiendo de $\|x_1,...,x_n,\alpha_1,...,\alpha_m\|$ y llega al estado $\|f(\vec{x},\vec{\alpha})\|$, o lo que es lo mismo, $\mathcal{P}$ termina partiendo de

$$\left\| x_1,...,x_n,\overbrace{0,...,0}^{k-n},\alpha_1,...,\alpha_m,\overbrace{\varepsilon,...,\varepsilon}^{k-m} \right\|$$

y llega al estado

$$\left\| \overbrace{0,...,0}^{k}, f(\vec{x},\vec{\alpha}),\overbrace{\varepsilon,...,\varepsilon}^{k-1} \right\|$$

Pero entonces el Lema 114 nos dice que

$$(***) \quad \left\lfloor q_0 B \shortmid^{x_1} B...B \shortmid^{x_n} B^{k-n} B\alpha_1 B...B\alpha_m B \right\rfloor \overset{*}{\underset{M_{sim}}{\vdash}} \left\lfloor q_f B^{k+1} f(\vec{x},\vec{\alpha}) \right\rfloor$$

Como ya lo vimos, si hacemos funcionar a M desde $\lfloor q_0 B \shortmid^{x_1} B...B \shortmid^{x_n} B\alpha_1 B...B\alpha_m B \rfloor$, llegaremos (via la copia de M_1 dentro de M) indefectiblemente a la siguiente descripcion instantanea

$$\left\lfloor q_2 B \shortmid^{x_1} B...B \shortmid^{x_n} B^{k-n} B\alpha_1 B...B\alpha_m B \right\rfloor$$

Luego (***) nos dice que, via la copia de M_{sim} dentro de M, llegaremos a $\left\lfloor q_3 B^{k+1} f(\vec{x},\vec{\alpha}) \right\rfloor$ e inmediatamente a $\left\lfloor q_4 B^{k+1} f(\vec{x},\vec{\alpha}) \right\rfloor$. Finalmente, via la copia de M_2 dentro de M, llegaremos a $\lfloor q_5 B f(\vec{x},\vec{\alpha}) \rfloor$, lo cual termina de demostrar que M computa a f $\blacksquare$

3.5 Conclusiones: La tesis de Church

En virtud de los teoremas ya probados tenemos el siguiente teorema que asegura que los tres paradigmas son equivalentes.

Theorem 117 *Sea Σ un alfabeto finito. Dada una funcion f, las siguientes son equivalentes:*

(1) f es Σ-Turing computable

(2) f es Σ-recursiva

(3) f es Σ-computable

Proof. (1)$\Rightarrow$(2) es probado en el Teorema 113. (2)$\Rightarrow$(3) es probado en el Teorema 90. (3)$\Rightarrow$(1) es probado en el Teorema 116. ■

Tambien los tres paradigmas son equivalentes con respecto a los dos tipos de conjuntos estudiados, es decir:

Theorem 118 *Sea Σ un alfabeto finito y sea $S \subseteq \omega^n \times \Sigma^{*m}$. Las siguientes son equivalentes:*

(1) S es Σ-Turing enumerable

(2) S es Σ-recursivamente enumerable

(3) S es Σ-enumerable

Proof. Directo de las definiciones y el teorema anterior. ■

Theorem 119 *Sea Σ un alfabeto finito y sea $S \subseteq \omega^n \times \Sigma^{*m}$. Las siguientes son equivalentes:*

(1) S es Σ-Turing computable

(2) S es Σ-recursivo

(3) S es Σ-computable

Proof. Directo de las definiciones y el teorema anterior. ■

Otro modelo matematico de computabilidad efectiva es el llamado lamda calculus, introducido por Church, el cual tambien resulta equivalente a los estudiados por nosotros. El hecho de que tan distintos paradigmas computacionales hayan resultado equivalentes hace pensar que en realidad los mismos han tenido exito en capturar la totalidad de las funciones Σ-efectivamente computables. Esta aseveracion es conocida como la

Tesis de Church: *Toda funcion Σ-efectivamente computable es Σ-recursiva.*

Si bien no se ha podido dar una prueba estrictamente matematica de la Tesis de Church, es un sentimiento comun de los investigadores del area que la misma es verdadera.

3.6 Resultados basicos presentados en paradigma recursivo

En esta seccion presentaremos varios de los resultados basicos de computabilidad, expresados en el paradigma recursivo, ya que es el mas habitual y comodo. Varios de estos resultados ya han sido establecidos dentro del desarrollo de la computabilidad efectiva en el Capitulo 2. A estos resultados los enunciaremos dentro del paradigma de Godel y daremos pruebas rigurosas matematicas de ellos usando la teoria desarrollada hasta ahora. Sin envargo, veremos que hay otros resultados que son dependientes del desarrollo matematico hecho y aportan nueva informacion al paradigma filosofico (la indecidibilidad del halting problem, por ejemplo).

3.6.1 Lema de division por casos para funciones Σ-recursivas

Lemma 120 *Supongamos $f_i : D_{f_i} \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$, $i = 1, ..., k$, son funciones Σ-recursivas tales que $D_{f_i} \cap D_{f_j} = \emptyset$ para $i \neq j$. Entonces la funcion $f_1 \cup ... \cup f_k$ es Σ-recursiva.*

Proof. Probaremos el caso $k = 2$ y $O = \Sigma^*$. Ademas supondremos que $n = m = 1$. Sean $\mathcal{P}_1$ y $\mathcal{P}_2$ programas que computen las funciones f_1 y f_2, respectivamente. Para $i = 1, 2$, definamos

$$H_i = \lambda t x_1 \alpha_1 \left[Halt^{1,1}(t, x_1, \alpha_1, \mathcal{P}_i) \right]$$

Notar que $D_{H_i} = \omega^2 \times \Sigma^*$ y que H_i es Σ-mixta. Ademas sabemos que la funcion $Halt^{1,1}$ es $(\Sigma \cup \Sigma_p)$-p.r. por lo cual resulta facilmente que H_i es $(\Sigma \cup \Sigma_p)$-p.r.. Por el Teorema de Independencia del Alfabeto tenemos que H_i es Σ-p.r.. Entonces H_i es Σ-computable por lo cual tenemos que hay un macro:

$$[\text{IF } H_i(\text{V1}, \text{V2}, \text{W1}) \text{ GOTO A1}]$$

Para hacer mas intuitivo el uso de este macro lo escribiremos de la siguiente manera

$$\left[\text{IF } Halt^{1,1}(\text{V1}, \text{V2}, \text{W1}, \mathcal{P}_i) \text{ GOTO A1}\right]$$

Ya que cada f_i es Σ-computable, hay macros

$$[\text{W2} \leftarrow f_1(\text{V1}, \text{W1})]$$
$$[\text{W2} \leftarrow f_2(\text{V1}, \text{W1})]$$

Sea $\mathcal{P}$ el siguiente programa:

$$\begin{array}{l}
\text{L1 N20} \leftarrow \text{N20} + 1 \\
\left[\text{IF } Halt^{1,1}(\text{N20}, \text{N1}, \text{P1}, \mathcal{P}_1) \text{ GOTO L2}\right] \\
\left[\text{IF } Halt^{1,1}(\text{N20}, \text{N1}, \text{P1}, \mathcal{P}_2) \text{ GOTO L3}\right] \\
\text{GOTO L1} \\
\text{L2 } [\text{P1} \leftarrow f_1(\text{N1}, \text{P1})] \\
\text{GOTO L4} \\
\text{L3 } [\text{P1} \leftarrow f_2(\text{N1}, \text{P1})] \\
\text{L4 SKIP}
\end{array}$$

Notese que $\mathcal{P}$ computa la funcion $f_1 \cup f_2$ ■

La prueba del lema anterior es de naturaleza imperativa ya que da explicitamente un programa (de todas maneras usa el paradigma recursivo o Godeliano para justificar la existencia de los macros). A continuacion daremos una prueba la cual es mas recursiva (aunque aun usa el paradigma imperativo en la existencia de los programas $\mathcal{P}_i$).

Proof. Sean $\mathcal{P}_1$ y $\mathcal{P}_2$ programas que computen las funciones f_1 y f_2, respectivamente. Sean

$$\begin{array}{rcl} P_1 & = & \lambda t\vec{x}\vec{\alpha}\left[Halt^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}_1)\right] \\ P_2 & = & \lambda t\vec{x}\vec{\alpha}\left[Halt^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}_2)\right] \end{array}$$

Notese que $D_{P_1} = D_{P_2} = \omega \times \omega^n \times \Sigma^{*m}$ y que P_1 y P_2 son $(\Sigma \cup \Sigma_p)$-p.r.. Ya que son Σ-mixtos, el Teorema 84 nos dice que son Σ-p.r.. Tambien notese que $D_{M((P_1 \vee P_2))} = D_{f_1} \cup D_{f_2}$. Definamos

$$\begin{array}{rcl} g_1 & = & \lambda\vec{x}\vec{\alpha}\left[E_{*1}^{n,m}(M\left((P_1 \vee P_2)\right)(\vec{x},\vec{\alpha}),\vec{x},\vec{\alpha},\mathcal{P}_1)^{P_1(M((P_1 \vee P_2))(\vec{x},\vec{\alpha}),\vec{x},\vec{\alpha})}\right] \\ g_2 & = & \lambda\vec{x}\vec{\alpha}\left[E_{*1}^{n,m}(M\left((P_1 \vee P_2)\right)(\vec{x},\vec{\alpha}),\vec{x},\vec{\alpha},\mathcal{P}_2)^{P_2(M((P_1 \vee P_2))(\vec{x},\vec{\alpha}),\vec{x},\vec{\alpha})}\right] \end{array}$$

Notese que g_1 y g_2 son Σ-recursivas y que $D_{g_1} = D_{g_2} = D_{f_1} \cup D_{f_2}$, Ademas notese que

$$g_1(\vec{x},\vec{\alpha}) = \begin{cases} f_1(\vec{x},\vec{\alpha}) & \text{si } (\vec{x},\vec{\alpha}) \in D_{f_1} \\ \varepsilon & \text{caso contrario} \end{cases}$$

$$g_2(\vec{x},\vec{\alpha}) = \begin{cases} f_2(\vec{x},\vec{\alpha}) & \text{si } (\vec{x},\vec{\alpha}) \in D_{f_2} \\ \varepsilon & \text{caso contrario} \end{cases}$$

O sea que $f_1 \cup f_2 = \lambda\alpha\beta\left[\alpha\beta\right] \circ [g_1, g_2]$ es Σ-recursiva. ■

3.6.2 Conjuntos Σ-recursivos y Σ-recursivamente enumerables

A continuacion probaremos los resultados basicos sobre conjuntos Σ-efectivamente computables y Σ-efectivamente enumerables, dados en las Secciones 2.3 y 2.2, pero enunciados dentro del paradigma de Godel.

Lemma 121 *Si $P : S \subseteq \omega^n \times \Sigma^{*m} \to \omega$ y $Q : S \subseteq \omega^n \times \Sigma^{*m} \to \omega$ son predicados Σ-r., entonces $(P \vee Q)$, $(P \wedge Q)$ y $\neg P$ lo son tambien.*

Proof. Note que

$$\begin{array}{rcl} \neg P & = & \lambda xy\left[x \dot{-} y\right] \circ [C_1^{n,m}, P] \\ (P \wedge Q) & = & \lambda xy\left[x.y\right] \circ [P, Q] \\ (P \vee Q) & = & \neg(\neg P \wedge \neg Q). \end{array}$$

■

Lemma 122 *Supongamos* $S_1, S_2 \subseteq \omega^n \times \Sigma^{*m}$ *son conjuntos* Σ*-recursivos. Entonces* $S_1 \cup S_2$*,* $S_1 \cap S_2$ *y* $S_1 - S_2$ *son* Σ*-recursivos*

Proof. Es directa del lema anterior. ■

Lemma 123 *Supongamos* $S_1, S_2 \subseteq \omega^n \times \Sigma^{*m}$ *son conjuntos* Σ*-r.e.. Entonces*

(1) $S_1 \cup S_2$ *es* Σ*-r.e..*

(2) $S_1 \cap S_2$ *es* Σ*-r.e..*

Proof. Podemos suponer que ni S_1 ni S_2 son vacios ya que de lo contrario los resultados son triviales. Ademas supondremos que $n = 2$ y $m = 1$.

(1). La idea de la prueba es la misma que la que usamos para probar que la union de conjuntos Σ-efectivamente enumerables es Σ-efectivamente enumerable. Daremos usando macros un programa que enumera a $S_1 \cup S_2$ y luego aplicaremos la Proposicion 89. Por hipotesis hay funciones $F : \omega \to \omega \times \omega \times \Sigma^*$ y $G : \omega \to \omega \times \omega \times \Sigma^*$ tales que $F_{(1)}, F_{(2)}, F_{(3)}, G_{(1)}, G_{(2)}$ y $G_{(3)}$ son Σ-recursivas, $\mathrm{Im}(F) = S_1$ y $\mathrm{Im}(G) = S_2$. Ya que estas funciones tambien son Σ-computables, hay macros

$$\begin{gathered}
\left[\mathrm{V}2 \leftarrow F_{(1)}(\mathrm{V}1)\right] \\
\left[\mathrm{V}2 \leftarrow F_{(2)}(\mathrm{V}1)\right] \\
\left[\mathrm{W}1 \leftarrow F_{(3)}(\mathrm{V}1)\right] \\
\left[\mathrm{V}2 \leftarrow G_{(1)}(\mathrm{V}1)\right] \\
\left[\mathrm{V}2 \leftarrow G_{(2)}(\mathrm{V}1)\right] \\
\left[\mathrm{W}1 \leftarrow G_{(3)}(\mathrm{V}1)\right]
\end{gathered}$$

Ya que el predicado $Par = \lambda x[x \text{ es par}]$ es Σ-p.r., tenemos que Par es Σ-computable. Es decir que hay un macro:

$$[\mathrm{IF}\ Par(\mathrm{V}1)\ \mathrm{GOTO\ A}1]$$

el cual escribiremos de la siguiente manera mas intuitiva

$$[\mathrm{IF\ V1\ es\ par\ GOTO\ A1}]$$

Ya que la funcion $D = \lambda x[\lfloor x/2 \rfloor]$ es Σ-p.r., tenemos que D es Σ-computable. Es decir que hay un macro:

$$[\mathrm{V}2 \leftarrow D(\mathrm{V}1)]$$

el cual escribiremos de la siguiente manera mas intuitiva

$$[\mathrm{V}2 \leftarrow \lfloor \mathrm{V}1/2 \rfloor]$$

Sea $\mathcal{P}$ el siguiente programa:

$$
\begin{array}{ll}
 & [\text{IF N1 es par GOTO L1} \\
 & \text{N1} \leftarrow \text{N1}\dot{-}1 \\
 & [\text{N1111} \leftarrow \lfloor \text{N1}/2 \rfloor] \\
 & \left[\text{N1} \leftarrow G_{(1)}(\text{N1111})\right] \\
 & \left[\text{N2} \leftarrow G_{(2)}(\text{N1111})\right] \\
 & \left[\text{P1} \leftarrow G_{(3)}(\text{N1111})\right] \\
 & \text{GOTO L2} \\
\text{L1} & [\text{N1111} \leftarrow \lfloor \text{N1}/2 \rfloor] \\
 & \left[\text{N1} \leftarrow F_{(1)}(\text{N1111})\right] \\
 & \left[\text{N2} \leftarrow F_{(2)}(\text{N1111})\right] \\
 & \left[\text{P1} \leftarrow F_{(3)}(\text{N1111})\right] \\
\text{L2} & \text{SKIP}
\end{array}
$$

Es facil ver que $\mathcal{P}$ cumple a y b de (3) de la Proposicion 89 por lo cual $S_1 \cup S_2$ es Σ-enumerable.

(2). Es dejada al lector ■

Tal como veremos mas adelante hay conjuntos Σ-recursivamente enumerables los cuales no son Σ-recursivos. Sin envargo tenemos el siguiente interesante resultado.

Theorem 124 *Sea $S \subseteq \omega^n \times \Sigma^{*m}$. Son equivalentes*

(a) S es Σ-recursivo

(b) S y $(\omega^n \times \Sigma^{*m}) - S$ son Σ-recursivamente enumerables

Proof. (a)$\Rightarrow$(b). Si $S = \emptyset$, por definicion S es Σ-recursivamente enumerable. Supongamos entonces $S \neq \emptyset$. Haremos el caso en el que $n = m = 1$ y $(0, \varepsilon) \in S$. Sea $\leq$ un orden total sobre Σ. Por hipotesis tenemos que $\chi_S^{\omega \times \Sigma^*}$ es Σ-recursiva por lo cual es Σ-computable. O sea que tenemos un macro

$$[\text{IF } \chi_S^{\omega \times \Sigma^*}(\text{V1}, \text{W1}) \text{ GOTO A1}]$$

Ya que la funcion $f = \lambda x[(x)_1]$ es Σ-p.r., ella es Σ-computable por lo cual hay un macro

$$[\text{V2} \leftarrow f(\text{V1})]$$

el cual escribiremos de la siguiente manera:

$$[\text{V2} \leftarrow (\text{V1})_1]$$

Ya que la funcion $g = \lambda x[*^{\leq}((x)_2)]$ es Σ-p.r., ella es Σ-computable por lo cual hay un macro

$$[\text{W1} \leftarrow g(\text{V1})]$$

el cual escribiremos de la siguiente manera:

$$[\mathrm{W1} \leftarrow *^{\leq}((\mathrm{V1})_2)]$$

(Dejamos al lector entender bien el funcionamiento de estos macros.) Sea $\mathcal{P}$ el siguiente programa:

$$\begin{array}{l} \mathrm{N1} \leftarrow \mathrm{N1}+1 \\ [\mathrm{N2} \leftarrow (\mathrm{N1})_1] \\ [\mathrm{P2} \leftarrow *^{\leq}(\mathrm{N1})_2] \\ [\mathrm{IF}\ \chi_S^{\omega\times\Sigma^*}(\mathrm{N2},\mathrm{P2})\ \mathrm{GOTO\ L1} \\ \mathrm{N1} \leftarrow 0 \\ \mathrm{P1} \leftarrow \varepsilon \\ \mathrm{GOTO\ L2} \\ \mathrm{L1}\ [\mathrm{N1} \leftarrow \mathrm{N2}] \\ [\mathrm{P1} \leftarrow \mathrm{P2})] \\ \mathrm{L2\ SKIP} \end{array}$$

Notese que $\mathrm{Dom}(\left[\Psi_{\mathcal{P}}^{1,0,\#},\Psi_{\mathcal{P}}^{1,0,*}\right]) = \omega$ y que $\mathrm{Im}(\left[\Psi_{\mathcal{P}}^{1,0,\#},\Psi_{\mathcal{P}}^{1,0,*}\right]) = S$ por lo cual S es Σ-enumerable lo que nos dice que S es Σ-recursivamente enumerable.

(b)$\Rightarrow$(a). Haremos el caso en que los conjuntos S y $(\omega^n \times \Sigma^{*m}) - S$ son no vacios. Tambien supondremos $n = m = 1$. Por hipotesis hay funciones $F : \omega \rightarrow \omega \times \Sigma^*$ y $G : \omega \rightarrow \omega \times \Sigma^*$ tales que $F_{(1)}$, $F_{(2)}$, $G_{(1)}$ y $G_{(2)}$ son Σ-recursivas, $\mathrm{Im}(F) = S$ y $\mathrm{Im}(G) = (\omega \times \Sigma^*) - S$. Ya que estas funciones tambien son Σ-computables, hay macros

$$\left[\mathrm{V2} \leftarrow F_{(1)}(\mathrm{V1})\right]$$
$$\left[\mathrm{W1} \leftarrow F_{(2)}(\mathrm{V1})\right]$$
$$\left[\mathrm{V1} \leftarrow G_{(1)}(\mathrm{V1})\right]$$
$$\left[\mathrm{W1} \leftarrow G_{(2)}(\mathrm{V1})\right]$$

Ya que los predicados $D = \lambda xy[x \neq y]$ y $D' = \lambda\alpha\beta[\alpha \neq \beta]$ son Σ-computables, hay macros

$$[\mathrm{IF}\ D(\mathrm{V1},\mathrm{V2})\ \mathrm{GOTO\ A1}]$$
$$[\mathrm{IF}\ D'(\mathrm{W1},\mathrm{W2})\ \mathrm{GOTO\ A1}]$$

los cuales para hacer mas amigable la lectura los escribieremos de la siguiente manera

$$[\mathrm{IF\ V1} \neq \mathrm{V2\ GOTO\ A1}]$$
$$[\mathrm{IF\ W1} \neq \mathrm{W2\ GOTO\ A1}]$$

Tambien usaremos el macro

$$[\mathrm{V1} \leftarrow C_1^{0,0}(\Diamond)]$$

(asociado a la funcion Σ-computable $C_1^{0,0}$), el cual escribiremos de la siguiente manera

$$[\mathrm{V1} \leftarrow 1]$$

Sea $\mathcal{P}$ el siguiente programa:

L1 [N2 $\leftarrow F_{(1)}$(N20)]
[P2 $\leftarrow F_{(2)}$(N20)]
[IF N2 $\neq$ N1 GOTO L2]
[IF P2 $\neq$ P1 GOTO L2]
[N1 $\leftarrow$ 1]
GOTO L3
L2 [N2 $\leftarrow G_{(1)}$(N20)]
[P2 $\leftarrow G_{(2)}$(N20)]
[IF N2 $\neq$ N1 GOTO L4]
[IF P2 $\neq$ P1 GOTO L4]
N1 $\leftarrow$ 0
GOTO L3
L4 N20 $\leftarrow$ N20 + 1
GOTO L1
L3 SKIP

Notese que $\mathcal{P}$ computa a la funcion $\chi_S^{\omega\times\Sigma^*}$ por lo cual $\chi_S^{\omega\times\Sigma^*}$ es Σ-computable lo que nos dice que es Σ-recursiva. Esto por definicion nos dice que S es Σ-recursivo ■

Lemma 125 *Supongamos $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to O$ es Σ-recursiva y $S \subseteq D_f$ es Σ-r.e., entonces $f|_S$ es Σ-recursiva.*

Proof. Si $S = \emptyset$, entonces $f|_S = \emptyset$ y por lo tanto $f|_S$ es Σ-recursiva. Supongamos $S \neq \emptyset$. Haremos el caso $n = m = 1$ y $O = \Sigma^*$. Tenemos que hay una $F : \omega \to \omega \times \Sigma^*$ tal que $\operatorname{Im} F = S$ y $F_{(1)}$, $F_{(2)}$ son Σ-recursivas. Ya que f, $F_{(1)}$ y $F_{(2)}$ son Σ-computables, hay macros

$$[\mathrm{W2} \leftarrow f(\mathrm{V1}, \mathrm{W1})]$$
$$\left[\mathrm{V2} \leftarrow F_{(1)}(\mathrm{V1})\right]$$
$$\left[\mathrm{W1} \leftarrow F_{(2)}(\mathrm{V1})\right]$$

Usaremos los macros

$$[\text{IF V1} \neq \text{V2 GOTO A1}]$$
$$[\text{IF W1} \neq \text{W2 GOTO A1}]$$

Sea $\mathcal{P}$ el siguiente programa

$$\begin{array}{ll}
\text{L2} & [\text{N2} \leftarrow F_{(1)}(\text{N20})] \\
& [\text{P2} \leftarrow F_{(2)}(\text{N20})] \\
& [\text{IF N1} \neq \text{N2 GOTO L1}] \\
& [\text{IF P1} \neq \text{P2 GOTO L1}] \\
& [\text{P1} \leftarrow f(\text{N1}, \text{P1})] \\
& \text{GOTO L3} \\
\text{L1} & \text{N20} \leftarrow \text{N20} + 1 \\
& \text{GOTO L2} \\
\text{L3} & \text{SKIP}
\end{array}$$

Es facil ver que $\mathcal{P}$ computa a $f|_S$ ■

Ahora probaremos el analogo recursivo del Teorema 31.

Theorem 126 *Dado $S \subseteq \omega^n \times \Sigma^{*m}$, son equivalentes*

(1) S es Σ-recursivamente enumerable

(2) $S = I_F$, para alguna $F : D_F \subseteq \omega^k \times \Sigma^{*l} \to \omega^n \times \Sigma^{*m}$ tal que cada $F_{(i)}$ es Σ-recursiva.

(3) $S = D_f$, para alguna funcion Σ-recursiva f

(4) $S = \emptyset$ o $S = I_F$, para alguna $F : \omega \to \omega^n \times \Sigma^{*m}$ tal que cada $F_{(i)}$ es Σ-p.r.

Proof. El caso $n = m = 0$ es facil y es dejado al lector. Supongamos entonces que $n + m \geq 1$.

(2)$\Rightarrow$(3). Haremos el caso $k = l = 1$ y $n = m = 2$. El caso general es completamente analogo. Notese que entonces tenemos que $S \subseteq \omega^2 \times \Sigma^{*2}$ y $F : D_F \subseteq \omega \times \Sigma^* \to \omega^2 \times \Sigma^{*2}$ es tal que $\operatorname{Im} F = S$ y $F_{(1)}$, $F_{(2)}$, $F_{(3)}$, $F_{(4)}$ son Σ-recursivas. Para cada $i \in \{1, 2, 3, 4\}$, sea $\mathcal{P}_i$ un programa el cual computa a $F_{(i)}$. Sea $\leq$ un orden total sobre Σ. Definamos

$$H_i = \lambda t x_1 \alpha_1 \left[\neg Halt^{1,1}(t, x_1, \alpha_1, \mathcal{P}_i)\right]$$

Notar que $D_{H_i} = \omega^2 \times \Sigma^*$ y que H_i es Σ-mixta. Ademas sabemos que la funcion $Halt^{1,1}$ es $(\Sigma \cup \Sigma_p)$-p.r. por lo cual resulta facilmente que H_i es $(\Sigma \cup \Sigma_p)$-p.r.. Por la Proposicion de Independencia del Alfabeto tenemos que H_i es Σ-p.r.. Entonces H_i es Σ-computable por lo cual tenemos que hay un macro:

$$[\text{IF } H_i(\text{V2}, \text{V1}, \text{W1}) \text{ GOTO A1}]$$

Para hacer mas intuitivo el uso de este macro lo escribiremos de la siguiente manera

$$\left[\text{IF } \neg Halt^{1,1}(\text{V2}, \text{V1}, \text{W1}, \mathcal{P}_i) \text{ GOTO A1}\right]$$

Para $i = 1, 2$, definamos

$$E_i = \lambda x t x_1 \alpha_1 \left[x \neq E^{1,1}_{\#1}(t, x_1, \alpha_1, \mathcal{P}_i) \right]$$

Para $i = 3, 4$, definamos

$$E_i = \lambda t x_1 \alpha_1 \alpha \left[\alpha \neq E^{1,1}_{*1}(t, x_1, \alpha_1, \mathcal{P}_i) \right]$$

Dejamos al lector probar que las funciones E_i son Σ-p.r.. O sea que son Σ-computables por lo cual para cada $i \in \{1, 2\}$ hay un macro

$$[\text{IF } E_i(\text{V2}, \text{V3}, \text{V1}, \text{W1}) \text{ GOTO A1}]$$

y para cada $i \in \{3, 4\}$ hay un macro

$$[\text{IF } E_i(\text{V2}, \text{V1}, \text{W1}, \text{W2}) \text{ GOTO A1}]$$

Haremos mas intuitiva la forma de escribir estos macros, por ejemplo para $i = 1$, lo escribiremos de la siguiente manera

$$\left[\text{IF V2} \neq E^{1,1}_{\#1}(\text{V3}, \text{V1}, \text{W1}, \mathcal{P}_1) \text{ GOTO A1} \right]$$

Ya que la funcion $f = \lambda x[(x)_1]$ es Σ-p.r., ella es Σ-computable por lo cual hay un macro

$$[\text{V2} \leftarrow f(\text{V1})]$$

el cual escribiremos de la siguiente manera:

$$[\text{V2} \leftarrow (\text{V1})_1]$$

Similarmente hay macros:

$$[\text{W1} \leftarrow *^{\leq}(\text{V1})_3]$$

$$[\text{V2} \leftarrow (\text{V1})_2]$$

(dejamos al lector entender bien el funcionamiento de estos macros). Sea $\mathcal{P}$ el siguiente programa de $\mathcal{S}^\Sigma$:

$$\begin{array}{l}
\text{L1 N20} \leftarrow \text{N20} + 1 \\
[\text{N10} \leftarrow (\text{N20})_1] \\
[\text{N3} \leftarrow (\text{N20})_2] \\
[\text{P3} \leftarrow *^{\leq}(\text{N20})_3] \\
\left[\text{IF } \neg Halt^{1,1}(\text{N10}, \text{N3}, \text{P3}, \mathcal{P}_1) \text{ GOTO L1} \right] \\
\left[\text{IF } \neg Halt^{1,1}(\text{N10}, \text{N3}, \text{P3}, \mathcal{P}_2) \text{ GOTO L1} \right] \\
\left[\text{IF } \neg Halt^{1,1}(\text{N10}, \text{N3}, \text{P3}, \mathcal{P}_3) \text{ GOTO L1} \right] \\
\left[\text{IF } \neg Halt^{1,1}(\text{N10}, \text{N3}, \text{P3}, \mathcal{P}_4) \text{ GOTO L1} \right] \\
\left[\text{IF N1} \neq E^{1,1}_{\#1}(\text{N10}, \text{N3}, \text{P3}, \mathcal{P}_1) \text{ GOTO L1} \right] \\
\left[\text{IF N2} \neq E^{1,1}_{\#1}(\text{N10}, \text{N3}, \text{P3}, \mathcal{P}_2) \text{ GOTO L1} \right] \\
\left[\text{IF P1} \neq E^{1,1}_{*1}(\text{N10}, \text{N3}, \text{P3}, \mathcal{P}_3) \text{ GOTO L1} \right] \\
\left[\text{IF P2} \neq E^{1,1}_{*1}(\text{N10}, \text{N3}, \text{P3}, \mathcal{P}_4) \text{ GOTO L1} \right]
\end{array}$$

Dejamos al lector la tarea de comprender el funcionamiento de este programa y convenserse de que computa la funcion $p_1^{2,2}|_S$. Pero entonces $p_1^{2,2}|_S$ es Σ-computable por lo cual es Σ-recursiva, lo cual prueba (3) ya que $\mathrm{Dom}(p_1^{2,2}|_S) = S$.

(3)$\Rightarrow$(4). Supongamos $S \neq \emptyset$. Sea $(z_1,...,z_n,\gamma_1,...,\gamma_m) \in S$ fijo. Sea $\mathcal{P}$ un programa el cual compute a f y Sea $\leq$ un orden total sobre Σ. Sea $P : \mathbf{N} \rightarrow \omega$ dado por $P(x) = 1$ sii

$$Halt^{n,m}\left((x)_{n+m+1},(x)_1,...,(x)_n,*^{\leq}((x)_{n+1}),...,*^{\leq}((x)_{n+m})),\mathcal{P}\right) = 1$$

Es facil ver que P es $(\Sigma \cup \Sigma_p)$-p.r. por lo cual es Σ-p.r.. Sea $\bar{P} = P \cup C_0^{1,0}|_{\{0\}}$. Para $i = 1,...,n$, definamos $F_i : \omega \rightarrow \omega$ de la siguiente manera

$$F_i(x) = \begin{cases} (x)_i & \text{si} \quad \bar{P}(x) = 1 \\ z_i & \text{si} \quad \bar{P}(x) \neq 1 \end{cases}$$

Para $i = n+1,...,n+m$, definamos $F_i : \omega \rightarrow \Sigma^*$ de la siguiente manera

$$F_i(x) = \begin{cases} *^{\leq}((x)_i) & \text{si} \quad \bar{P}(x) = 1 \\ \gamma_{i-n} & \text{si} \quad \bar{P}(x) \neq 1 \end{cases}$$

Por el lema de division por casos, cada F_i es Σ-p.r.. Es facil ver que $F = [F_1,...,F_{n+m}]$ cumple (4). ■

La prueba de (2)$\Rightarrow$(3) del teorema anterior es de naturaleza imperativa ya que da explicitamente un programa (de todas maneras usa el paradigma recursivo o Godeliano para justificar la existencia de los macros). A continuacion daremos una prueba de (2)$\Rightarrow$(3) la cual es mas recursiva (aunque aun usa el paradigma imperativo en la existencia de los programas $\mathcal{P}_i$).

(2)$\Rightarrow$(3). Para $i = 1,...,n+m$, sea $\mathcal{P}_i$ un programa el cual computa a $F_{(i)}$ y Sea $\leq$ un orden total sobre Σ. Sea $P : \mathbf{N} \times \omega^n \times \Sigma^{*m} \rightarrow \omega$ dado por $P(t,\vec{x},\vec{\alpha}) = 1$ sii se cumplen las siguientes condiciones

$$\begin{array}{rcl}
Halt^{k,l}(\left((t)_{k+l+1},(t)_1,...,(t)_k,*^{\leq}((t)_{k+1}),...,*^{\leq}((t)_{k+l})),\mathcal{P}_1\right) & = & 1 \\
 & & \vdots \\
Halt^{k,l}\left((t)_{k+l+1},(t)_1...(t)_k,*^{\leq}((t)_{k+1})...*^{\leq}((t)_{k+l})),\mathcal{P}_{n+m}\right) & = & 1 \\
E_{\#1}^{k,l}((t)_{k+l+1},(t)_1,...,(t)_k,*^{\leq}((t)_{k+1}),...,*^{\leq}((t)_{k+l})),\mathcal{P}_1) & = & x_1 \\
 & & \vdots \\
E_{\#1}^{k,l}((t)_{k+l+1},(t)_1,...,(t)_k,*^{\leq}((t)_{k+1}),...,*^{\leq}((t)_{k+l})),\mathcal{P}_n) & = & x_n \\
E_{*1}^{k,l}((t)_{k+l+1},(t)_1,...,(t)_k,*^{\leq}((t)_{k+1}),...,*^{\leq}((t)_{k+l})),\mathcal{P}_{n+1}) & = & \alpha_1 \\
 & & \vdots \\
E_{*1}^{k,l}((t)_{k+l+1},(t)_1,...,(t)_k,*^{\leq}((t)_{k+1}),...,*^{\leq}((t)_{k+l})),\mathcal{P}_{n+m}) & = & \alpha_m
\end{array}$$

Note que P es $(\Sigma \cup \Sigma_p)$-p.r. y por lo tanto P es Σ-p.r.. Pero entonces $M(P)$ es Σ-r. lo cual nos dice que se cumple (3) ya que $D_{M(P)} = I_F = S$. ■

Corollary 127 *Supongamos $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to O$ es Σ-recursiva y $S \subseteq I_f$ es Σ-r.e., entonces $f^{-1}(S) = \{(\vec{x}, \vec{\alpha}) : f(\vec{x}, \vec{\alpha}) \in S\}$ es Σ-r.e..*

Proof. Por el teorema anterior $S = D_g$, para alguna funcion Σ-recursiva g. Note que $f^{-1}(S) = D_{g \circ f}$, lo cual nuevamente por el teorema anterior nos dice que $f^{-1}(S)$ es Σ-r.e.. ■

Dejamos como ejercicio dar una prueba imperativa del corolario anterior. Los Lemas 125 y 123 pueden obtenerse facilmente como corolarios del teorema anterior. Se gana en elegancia y simplicidad pero cabe destacar que se pierde en intuicion

Corollary 128 *Supongamos $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to O$ es Σ-recursiva y $S \subseteq D_f$ es Σ-r.e., entonces $f|_S$ es Σ-recursiva.*

Proof. Supongamos $O = \Sigma^*$. Por el teorema anterior $S = D_g$, para alguna funcion Σ-recursiva g. Notese que componiendo adecuadamente podemos suponer que $I_g = \{\varepsilon\}$. O sea que tenemos $f|_S = \lambda\alpha\beta\,[\alpha\beta] \circ [f, g]$. ■

Corollary 129 *Supongamos $S_1, S_2 \subseteq \omega^n \times \Sigma^{*m}$ son conjuntos Σ-r.e.. Entonces $S_1 \cap S_2$ es Σ-r.e..*

Proof. Por el teorema anterior $S_i = D_{g_i}$, con g_1, g_2 funciones Σ-recursivas. Notese que podemos suponer que $I_{g_1}, I_{g_2} \subseteq \omega$ por lo que $S_1 \cap S_2 = D_{\lambda xy[xy] \circ [g_1, g_2]}$ es Σ-r.e.. ■

Corollary 130 *Supongamos $S_1, S_2 \subseteq \omega^n \times \Sigma^{*m}$ son conjuntos Σ-r.e.. Entonces $S_1 \cup S_2$ es Σ-r.e.*

Proof. Supongamos $S_1 \neq \emptyset \neq S_2$. Sean $F, G : \omega \to \omega^n \times \Sigma^{*m}$ tales que $I_F = S_1$, $I_G = S_2$ y las funciones $F_{(i)}$'s y $G_{(i)}$'s son Σ-recursivas. Sean $f = \lambda x\,[Q(x, 2)]$ y $g = \lambda x\,\left[Q(x \dot{-} 1, 2)\right]$. Sea $H : \omega \to \omega^n \times \Sigma^{*m}$ dada por

$$H_{(i)} = (F_{(i)} \circ f)|_{\{x:x \text{ es par}\}} \cup (G_{(i)} \circ g)|_{\{x:x \text{ es impar}\}}$$

Por el Lema 125 y el Lema 120, cada H_i es Σ-recursiva. Ya que $I_H = S_1 \cup S_2$.tenemos que $S_1 \cup S_2$ es Σ-r.e. ■

A continuacion dejamos un sketch de una prueba alternativa del Teorema 124. Dejamos al lector completar los detalles.

Proof. (a)$\Rightarrow$(b). Note que $S = D_{Pred \circ \chi_S^{\omega^n \times \Sigma^{*m}}}$.

(b)$\Rightarrow$(a). Note que $\chi_S^{\omega^n \times \Sigma^{*m}} = C_1^{n,m}|_S \cup C_0^{n,m}|_{(\omega^n \times \Sigma^{*m}) - S}$. ■

Los dos siguientes teoremas, nos agregan una equivalencia mas al Teorema 126, para el caso $n = 0$, $m = 1$.

Theorem 131 *Si $L \subseteq \Sigma^*$ es Σ-r.e., entonces $L = L(M) = H(M)$ para alguna maquina de Turing deterministica M.*

Proof. La prueba es similar a la del Teorema 116 asique solo daremos un skech de la misma. Por el Teorema 126 $L = D_f$ para alguna funcion f la cual es Σ-recursiva. Notese que podemos suponer que $\operatorname{Im} f \subseteq \Sigma^*$. Ya que f es Σ-recursiva, tambien es Σ-computable. Por el Lema 115 existe $\mathcal{P} \in \mathrm{Pro}^\Sigma$ el cual computa f y tiene las propiedades (1) y (2). Sea $k = N(\mathcal{P})$ y sea M_{sim} la maquina de Turing con unit que simula a $\mathcal{P}$ respecto de k. Sea M_1 una maquina de Turing deterministica con un solo estado final q_f (del cual no salen flechas) y tal que para todo $\alpha \in \Sigma^*$,

$$\lfloor q_0 B\alpha \rfloor \overset{*}{\vdash} \lfloor q_f B^{k+1}\alpha \rfloor$$

Note que la concatenacion de M_1 con M produce una maquina de Turing deterministica M_2 tal que $H(M_2) = L(M_2) = L$. Dejamos al lector los detalles de la construccion de M_2. ■

Theorem 132 *Sea $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ una maquina de Turing. Entonces $L(M)$ y $H(M)$ son Σ-recursivamente enumerables.*

Proof. Veamos que $L(M)$ es Σ-recursivamente enumerable. Sea P el siguiente predicado $(\Gamma \cup Q)$-mixto

$$\lambda n\alpha \left[(\exists d \in Des) \ \lfloor q_0 B\alpha \rfloor \underset{M}{\overset{n}{\vdash}} d \wedge St(d) \in F \right]$$

Notese que $D_P = \omega \times \Gamma^*$. Dejamos al lector probar que P es $(\Gamma \cup Q)$-p.r.. Sea $P' = P|_{\omega \times \Sigma^*}$. Notese que $P'(n, \alpha) = 1$ sii $\alpha \in L(M)$ atestiguado por una computacion de longitud n. Ya que P' es $(\Gamma \cup Q)$-p.r. (por que?) y ademas es Σ-mixto, el Teorema 84 nos dice que P' es Σ-p.r.. Ya que $L(M) = D_{M(P')}$, el Teorema 126 nos dice que $L(M)$ es Σ-r.e..

Dejamos al lector la prueba parecida de que $H(M)$ es Σ-recursivamente enumerable. ■

3.6.3 El halting problem y los conjuntos A y N

Cuando $\Sigma \supseteq \Sigma_p$, podemos definir

$$AutoHalt^\Sigma = \lambda \mathcal{P} \left[(\exists t \in \omega) \ Halt^{0,1}(t, \mathcal{P}, \mathcal{P}) \right].$$

Notar que el dominio de $AutoHalt^\Sigma$ es Pro^Σ y que para cada $\mathcal{P} \in \mathrm{Pro}^\Sigma$ tenemos que

(*) $AutoHalt(\mathcal{P}) = 1$ sii $\mathcal{P}$ se detiene partiendo del estado $\|\mathcal{P}\|$.

Lemma 133 *Supongamos $\Sigma \supseteq \Sigma_p$. Entonces $AutoHalt^\Sigma$ no es Σ-recursivo.*

Proof. Supongamos $AutoHalt^{\Sigma}$ es Σ-recursivo y por lo tanto Σ-computable. Por la proposicion de existencia de macros tenemos que hay un macro

$$\left[\text{IF } AutoHalt^{\Sigma}(\text{W1}) \text{ GOTO A1}\right]$$

Sea $\mathcal{P}_0$ el siguiente programa de $\mathcal{S}^{\Sigma}$

$$\text{L1 } \left[\text{IF } AutoHalt^{\Sigma}(\text{P1}) \text{ GOTO L1}\right]$$

Note que

- $\mathcal{P}_0$ termina partiendo desde $\|\mathcal{P}_0\|$ sii $AutoHalt^{\Sigma}(\mathcal{P}_0) = 0$,

lo cual produce una contradiccion si tomamos en (*) $\mathcal{P} = \mathcal{P}_0$. ■

Usando el lema anterior y la Tesis de Church podemos probar el siguiente impactante resultado.

Theorem 134 *Supongamos $\Sigma \supseteq \Sigma_p$. Entonces $AutoHalt^{\Sigma}$ no es Σ-efectivamente computable. Es decir no hay ningun procedimiento efectivo que decida si un programa de $\mathcal{S}^{\Sigma}$ termina partiendo de si mismo.*

Proof. Si $AutoHalt^{\Sigma}$ fuera Σ-efectivamente computable, la Tesis de Church nos diria que es Σ-recursivo, contradiciendo el lema anterior. ■

Notese que $AutoHalt^{\Sigma}$ provee de un ejemplo natural en el cual la cuantificacion (no acotada) de un predicado Σ-p.r. con dominio rectangular no es Σ-efectivamente computable

Ahora estamos en condiciones de dar un ejemplo natural de un conjunto A que es Σ-recursivamente enumerable pero el cual no es Σ-recursivo.

Lemma 135 *Supongamos que $\Sigma \supseteq \Sigma_p$. Entonces*

$$A = \left\{\mathcal{P} \in \text{Pro}^{\Sigma} : AutoHalt^{\Sigma}(\mathcal{P}) = 1\right\}$$

es Σ-r.e. y no es Σ-recursivo. Mas aun el conjunto

$$N = \left\{\mathcal{P} \in \text{Pro}^{\Sigma} : AutoHalt^{\Sigma}(\mathcal{P}) = 0\right\}$$

no es Σ-r.e.

Proof. Para ver que A es Σ-r.e. se lo puede hacer imperativamente dando un programa (usando macros) que enumere a A. De esta forma probariamos que A es Σ-enumerable y por lo tanto es Σ-r.e.. Daremos ahora una prueba no imperativa de este hecho, es decir mas propia del paradigma funcional. Sea $P = \lambda t\mathcal{P}\left[Halt^{0,1}(t, \mathcal{P}, \mathcal{P})\right]$. Note que P es Σ-p.r. por lo que $M(P)$ es Σ-r.. Ademas note que $D_{M(P)} = A$, lo cual implica que A es Σ-r.e..

Supongamos ahora que N es Σ-r.e.. Entonces la funcion $C_0^{0,1}|_N$ es Σ-recursiva ya que $C_0^{0,1}$ lo es. Ademas ya que A es Σ-r.e. tenemos que $C_1^{0,1}|_A$ es Σ-recursiva. Ya que

$$AutoHalt^\Sigma = C_1^{0,1}|_A \cup C_0^{0,1}|_N$$

el Lema 120 nos dice que $AutoHalt^\Sigma$ es Σ-recursivo, contradiciendo el Lema 133. Esto prueba que N no es Σ-r.e..

Finalmente supongamos A es Σ-recursivo. Entonces el conjunto

$$N = (\Sigma^* - A) \cap \mathrm{Pro}^\Sigma$$

deberia serlo, lo cual es absurdo. Hemos probado entonces que A no es Σ-recursivo. ■

Cabe destacar aqui que el dominio de una funcion Σ-recursiva no siempre sera un conjunto Σ-recursivo. Por ejemplo si tomamos Σ tal que $\Sigma \supseteq \Sigma_p$, entonces $C_1^{0,1}|_A$ es una funcion Σ-recursiva ya que es la restriccion de la funcion Σ-recursiva $C_1^{0,1}$ al conjunto Σ-r.e. A, pero $\mathrm{Dom}(C_1^{0,1}|_A) = A$ no es Σ-recursivo.

Usando la Tesis de Church obtenemos el siguiente resultado.

Proposition 136 *Supongamos que $\Sigma \supseteq \Sigma_p$. Entonces A es Σ-efectivamente enumerable y no es Σ-efectivamente computable. El conjunto N no es Σ-efectivamente enumerable. Es decir, A puede ser enumerado por un procedimiento efectivo pero no hay ningun procedimiento efectivo que decida la pertenencia a A y no hay ningun procedimiento efectivo que enumere a N. Mas aun, si un procedimiento efectivo da como salida siempre elementos de N, entonces hay una cantidad infinita de elementos de N los cuales nunca da como salida*

Con los resultados anteriores estamos en condiciones de dar un ejemplo de un predicado Σ-recursivo, cuya minimizacion no es Σ-efectivamente computable (y por lo tanto es no Σ-recursiva).

Proposition 137 *Supongamos que $\Sigma \supseteq \Sigma_p$. Sea $P = C_1^{0,1}|_A \circ \lambda t\alpha \left[\alpha^{1 \dot{-} t}\mathrm{SKIP}^t\right]|_{\omega \times \mathrm{Pro}^\Sigma}$. La funcion $M(P)$ no es Σ-efectivamente computable (y por lo tanto es no Σ-recursiva)*

Proof. Notese que $D_{M(P)} = \mathrm{Pro}^\Sigma$ y que para cada $\mathcal{P} \in \mathrm{Pro}^\Sigma$ se tiene que

$$M(P)(\mathcal{P}) = 0 \text{ sii } \mathcal{P} \in A$$

O sea que $AutoHalt^\Sigma = \lambda x[x = 0] \circ M(P)$ lo cual nos dice que $M(P)$ no es Σ-recursiva ya que si lo fuese lo seria tambien $AutoHalt^\Sigma$. Por la Tesis de Church $M(P)$ tampoco es Σ-efectivamente computable ■

Supongamos $\Sigma \supseteq \Sigma_p$. Sea $f = \lambda\mathcal{P}\left[T^{0,1}(\mathcal{P},\mathcal{P})\right]$. Note que $D_f = A$ y $f(\mathcal{P})$ es la cantidad de pasos en la que $\mathcal{P}$ se detiene partiendo de $\|\mathcal{P}\|$.

Lemma 138 *No hay ninguna funcion* $F : \mathrm{Pro}^{\Sigma} \to \omega$ *la cual sea* Σ*-recursiva y extienda a* f

Proof. Supongamos hay una tal F. Notese que $AutoHalt^{\Sigma} = \lambda\mathcal{P}\left[Halt^{0,1}(F(\mathcal{P}),\mathcal{P},\mathcal{P})\right]$ lo cual nos dice que $AutoHalt^{\Sigma}$ es Σ-recursiva, llegando a una contradiccion. ■

4 Estructuras algebraicas ordenadas

En esta seccion estudiaremos varias clases de estructuras algebraicas en las cuales hay un orden parcial involucrado. Esto tendra una triple utilidad. Por un lado algunos de los resultados probados sobre algebras de Boole (por ejemplo el teorema de Rasiova y Sikorski) seran utilizados mas adelante para la prueba del teorema de completitud de la logica de primer orden. Tambien, esta seccion servira para volvernos algebristas maduros (lo mas que se pueda) ya que esto nos sera util a la hora de hacer logica matematica. La logica matematica es *matematica aplicada* al estudio de los matematicos, su lenguaje y sus metodos de demostracion, y que mas comodo para hacer logica matematica que contar con un matematico dentro de uno mismo para estudiarlo!

Finalmente cabe destacar que los resultados cubiertos en esta seccion son importantes en si mismos fuera de su vinculacion con la logica y tienen fuertes aplicaciones en otras disciplinas y ramas de la matematica.

4.1 Conjuntos parcialmente ordenados

Recordamos que tal como se lo definio en la Seccion 1.6.2, una relacion binaria $\leq$ sobre un conjunto P es llamada un *orden parcial sobre* P si se cumplen las siguientes condiciones:

(1) $\leq$ es reflexiva, i. e. para todo $a \in P$, $a \leq a$

(2) $\leq$ es antisimetrica, i. e. para todo $a, b \in P$, si $a \leq b$ y $b \leq a$, entonces $a = b$.

(3) $\leq$ es transitiva, i. e. para todo $a, b, c \in P$, si $a \leq b$ y $b \leq c$, entonces $a \leq c$.

(recomendamos antes de leer este tema, leer la Seccion 1.6.2, para familiarizarse con la notacion y las propiedades basicas de los ordenes parciales).

Un *conjunto parcialmente ordenado o poset* es un par $(P, \leq)$ donde P es un conjunto no vacio cualquiera y $\leq$ es un orden parcial sobre P. Dado un poset $(P, \leq)$, el conjunto P sera llamado el *universo* de $(P, \leq)$. Algunos ejemplos:

(E1) $(\mathbf{R}, \leq)$ es un poset, donde $\leq$ es el orden usual de los numeros relales

(E2) $(\{1,2,3\}, \{(1,2),(1,3),(1,1),(2,2),(3,3)\})$ es un poset

(E3) $(\mathcal{P}(\omega), \leq)$ es un poset, donde $\leq\, = \{(S,T) \in \mathcal{P}(\omega)^2 : S \subseteq T\}$

(E4) $(\{1\}, \{(1,1)\}$ es un poset

(E5) $(\mathbf{N}, \leq)$ es un poset, donde $\leq = \{(n,m) \in \mathbf{N}^2 : n \mid m\}$

(E6) $(A, \{(a,b) : a = b\})$ es un poset, cualesquiera sea el conjunto no vacio A

Usaremos la siguiente

Convencion notacional 1 Si hemos denotado con $\leq$ a cierto orden parcial sobre un conjunto A, entonces

(a) Denotaremos con $<$ a la relacion binaria $\{(a,b) \in A^2 : a \leq b \text{ y } a \neq b\}$. Es decir que $< = \{(a,b) \in A^2 : a \leq b \text{ y } a \neq b\}$. Cuando se de $a < b$ diremos que *a es menor que b* o que *b es mayor que a* (*respecto de* $\leq$)

(b) Denotaremos con $\prec$ a la relacion binaria

$$\{(a,b) \in A^2 : a < b \text{ y no existe } z \text{ tal que } a < z < b\}$$

Cuando se de $a \prec b$ diremos que *a es cubierto por b* o que *b cubre a a* (*respecto de* $\leq$).

El mismo tipo de convencion notacional se hara cuando denotemos con $\leq'$ (o $\tilde{\leq}$, etc) a un orden parcial sobre A. Es decir tendremos dos relaciones binarias nuevas tacitamente definidas, a saber:

$$< \quad ' = \{(a,b) \in A^2 : a \leq' b \text{ y } a \neq b\}$$
$$\prec \quad ' = \{(a,b) \in A^2 : a <' b \text{ y no existe } z \text{ tal que } a <' z <' b\}$$

4.1.1 Diagramas de Hasse

Dado un poset $(P, \leq)$, con P un conjunto finito, podemos realizar un diagrama llamado *diagrama de Hasse,* siguiendo las siguientes instrucciones:

(1) Asociar en forma inyectiva, a cada $a \in P$ un punto p_a del plano

(2) Trazar un segmento de recta uniendo los puntos p_a y p_b, cada vez que $a \prec b$

(3) Realizar lo indicado en los puntos (1) y (2) en tal forma que

(i) Si $a \prec b$, entonces p_a esta por debajo de p_b

(ii) Si un punto p_a ocurre en un segmento del diagrama entonces lo hace en alguno de sus extremos.

La relacion de orden $\leq$ puede ser facilmente obtenida a partir del diagrama, a saber, $a \leq b$ sucedera si y solo si $p_a = p_b$ o hay una sucesion de segmentos ascendentes desde p_a hasta p_b.

Algunos ejemplos:

4.1.2 Elementos maximales, maximos, minimales y minimos

Sea $(P, \leq)$ un poset. Diremos que $a \in P$ es un elemento *maximal de* $(P, \leq)$ si no existe un $b \in P$ tal que $a < b$. Diremos que $a \in P$ es un elemento *maximo de* $(P, \leq)$ si $b \leq a$, para todo $b \in P$. En forma analoga se definen los conceptos de elemento *minimal* y *minimo*. Algunos ejemplos:

(E1) Sea $\leq$ el orden usual de los numeros reales. El poset $(\mathbf{R}, \leq)$ no tiene elemento maximo ni minimo. Tampoco tiene elementos maximales ni minimales.

(E2) El poset $\mathbf{P} = (\{1, 2, 3\}, \{(1, 2), (1, 3), (1, 1), (2, 2), (3, 3)\})$ no tiene elemento maximo. 1 es un elemento minimo de $\mathbf{P}$. El unico elemento minimal de $\mathbf{P}$ es 1. Los elementos 2 y 3 son los unicos maximales de $\mathbf{P}$.

(E3) 1 es un elemento maximo de del poset $(\{1\}, \{(1, 1)\})$. Tambien 1 es un elemento minimo de $(\{1\}, \{(1, 1)\})$.

Como lo muestra el ejemplo (E3), no siempre hay elementos maximales o maximos en un poset. Ademas un poset tiene a lo sumo un maximo y un minimo (por que?), los cuales en caso de existir algunas veces seran denotados con 1 y 0, respectivamente. Tambien diremos que $(P, \leq)$ *tiene un* 1 (resp. 0) para expresar que $(P, \leq)$ tiene un elemento maximo (resp. minimo). Notese tambien que todo elemento maximo (resp. minimo) de $(P, \leq)$ es un elemento maximal (resp. minimal) de $(P, \leq)$ (por que?).

4.1.3 Supremos

Sea $(P, \leq)$ un poset. Dado $S \subseteq P$, diremos que un elemento $a \in P$ es *cota superior de* S *en* $(P, \leq)$ cuando $b \leq a$, para todo $b \in S$. Notese que todo elemento de P es cota superior de $\emptyset$ en $(P, \leq)$. Un elemento $a \in P$ sera llamado *supremo de* S *en* $(P, \leq)$ cuando se den las siguientes dos propiedades

(1) a es a cota superior de S en $(P, \leq)$

(2) Para cada $b \in P$, si b es una cota superior de S en $(P, \leq)$, entonces $a \leq b$.

Algunos ejemplos:

(E1) Consideremos el poset $(\mathbf{R}, \leq)$, donde $\leq$ es el orden usual de los numeros reales. Notese que ningun elemento de $\mathbf{R}$ es cota superior de ω en $(\mathbf{R}, \leq)$. O sea que ningun elemento de $\mathbf{R}$ es supremo de ω en $(\mathbf{R}, \leq)$. Sea

$$\begin{aligned} S &= \{-1/n : n \in \mathbf{N}\} \\ &= \{-1, -1/2, -1/3, ...\} \end{aligned}$$

Es facil ver que 0 es supremo de S en $(\mathbf{R}, \leq)$.

(E2) Consideremos el poset $(\mathcal{P}(\omega), \leq)$, donde $\leq = \{(A,B) \in \mathcal{P}(\omega)^2 : A \subseteq B\}$. Sean $A, B \in \mathcal{P}(\omega)$. Es facil ver que $A \cup B$ es supremo de $\{A, B\}$ en $(\mathcal{P}(\omega), \leq)$.

Como lo muestra el ejemplo (E1) no siempre existe un supremo de S en $(P, \leq)$. Ademas notese que en caso de existir es unico, es decir, si a es supremo de S en $(P, \leq)$ y a' es supremo de S en $(P, \leq)$, entonces $a = a'$ (por que?). Esto nos permite hablar de EL supremo de S en $(P, \leq)$, cuando exista. Denotaremos con $\sup(S)$ al supremo de S en $(P, \leq)$, en caso de que exista. Si en algun enunciado nuestro aparece la expresion $\sup(S)$, implicitamente ya estaremos aceptando que existe el supremo de S en $(P, \leq)$.

Notese que (E1) nos muestra que no siempre el supremo de un conjunto pertenece al conjunto. Notese ademas que, en caso de existir, el supremo del conjunto $\emptyset$ en $(P, \leq)$ es un elemento minimo de $(P, \leq)$. Esto es porque todo elemento de P es cota superior de $\emptyset$ en $(P, \leq)$.

Daremos otro ejemplo muy importante pero antes un poco de matematica basica. Recordemos que dados $x, y \in \mathbf{N}$ decimos que *x es multiplo de y* cuando y divide a x. Ademas, por definicion, el *minimo comun multiplo de x e y* es el menor elemento del conjunto $\{z \in \mathbf{N} : z \text{ es multiplo de } x \text{ y de } y\}$. El minimo comun multiplo de x e y se denota con $mcm(x, y)$. Una propiedad importante es la siguiente:

(*) Si z es multiplo de x y de y, entonces $mcm(x, y)|z$, es decir no solo $mcm(x, y)$ es menor o igual a cada multiplo comun de x e y, sino que $mcm(x, y)$ divide a cada multiplo comun de x e y

Un ejemplo importante de existencia de supremos es el siguiente:

(E3) Consideremos el poset $(\mathbf{N}, D)$, donde $D = \{(x, y) \in \mathbf{N}^2 : x|y\}$. Dados $x, y \in \mathbf{N}$, se tiene que $mcm(x, y)$ es el supremo de $\{x, y\}$ en $(\mathbf{N}, D)$. Es claro que $mcm(x, y)$ es cota superior de $\{x, y\}$ en $(\mathbf{N}, D)$. Ademas la propiedad (*) nos asegura que la propiedad (2) de la definicion de supremo se cumple. No es obvio que se cumple (2) de la definicion de supremo? Por que es necesario aplicar la propiedad (*)?

4.1.4 Infimos

Sea $(P, \leq)$ un poset. Dado $S \subseteq P$, diremos que un elemento $a \in P$ es *cota inferior de S en* $(P, \leq)$, cuando $a \leq b$, para cada $b \in S$. Notese que todo elemento de P es cota inferior de $\emptyset$ en $(P, \leq)$. Un elemento $a \in P$ sera llamado *infimo de S en* $(P, \leq)$ cuando se den las siguientes dos propiedades

(1) a es a cota inferior de S en $(P, \leq)$

(2) Para cada $b \in P$, si b es una cota inferior de S en $(P, \leq)$, entonces $b \leq a$.

Algunos ejemplos:

(E1) Consideremos el poset $(\mathbf{R}, \leq)$, donde $\leq$ es el orden usual de los numeros reales. Notese que ningun elemento de $\mathbf{R}$ es cota inferior de $\mathbf{Z}$ en $(\mathbf{R}, \leq)$. O sea que ningun elemento de $\mathbf{R}$ es infimo de $\mathbf{Z}$ en $(\mathbf{R}, \leq)$. Sea

$$\begin{aligned} S &= \{1/n : n \in \mathbf{N}\} \\ &= \{1, 1/2, 1/3, ...\} \end{aligned}$$

Es facil ver que 0 es infimo de S en $(\mathbf{R}, \leq)$. Notar que $\inf S \notin S$.

(E2) Consideremos el poset $(\mathcal{P}(\omega), \leq)$, donde $\leq = \{(A, B) \in \mathcal{P}(\omega)^2 : A \subseteq B\}$. Sean $A, B \in \mathcal{P}(\omega)$. Es facil ver que $A \cap B$ es infimo de $\{A, B\}$ en $(\mathcal{P}(\omega), \leq)$.

Como lo muestra el ejemplo (E1) no siempre existe un infimo de S en $(P, \leq)$. Ademas notese que en caso de existir es unico, es decir, si a es infimo de S en $(P, \leq)$ y a' es infimo de S en $(P, \leq)$, entonces $a = a'$ (por que?). Esto nos permite hablar de EL infimo de S en $(P, \leq)$, cuando exista. Denotaremos con $\inf(S)$ al infimo de S en $(P, \leq)$, en caso de que exista. Si en algun enunciado nuestro aparece la expresion $\inf(S)$, implicitamente ya estaremos aceptando que existe el infimo de S en $(P, \leq)$.

Notese que (E1) nos muestra que no siempre el infimo de un conjunto pertenece al conjunto. Notese ademas que en caso de existir el infimo del conjunto $\emptyset$ en $(P, \leq)$ es un elemento maximo de $(P, \leq)$.

Daremos otro ejemplo muy importante pero antes un poco de matematica basica. Recordemos que dados $x, y \in \mathbf{N}$, por definicion, el *maximo comun divisor de x e y* es el mayor elemento del conjunto $\{z \in \mathbf{N} : z|x \text{ y } z|y\}$. El maximo comun divisor de x e y se denota con $mcd(x, y)$. Una propiedad importante es la siguiente:

(**) Si $z|x$ y $z|y$, entonces $z|mcd(x, y)$, es decir no solo $mcd(x, y)$ es mayor o igual a cada divisor comun de x e y, sino que $mcd(x, y)$ es divisible por cada divisor comun de x e y

Un ejemplo importante de existencia de infimos es el siguiente:

(E3) Consideremos el poset $(\mathbf{N}, D)$, donde $D = \{(x, y) \in \mathbf{N}^2 : x|y\}$. Dados $x, y \in \mathbf{N}$, se tiene que $mcd(x, y)$ es el infimo de $\{x, y\}$ en $(\mathbf{N}, D)$. Es claro que $mcd(x, y)$ es cota inferior de $\{x, y\}$ en $(\mathbf{N}, D)$. Ademas la propiedad (**) nos asegura que la propiedad (2) de la definicion de infimo se cumple. No es obvio que se cumple (2) de la definicion de infimo? Por que es necesario aplicar la propiedad (**)?

4.1.5 Homomorfismos de posets

Sean $(P, \leq)$ y $(P', \leq')$ posets. Una funcion $F : P \to P'$ sera llamada un *homomorfismo de* $(P, \leq)$ *en* $(P', \leq')$ si para todo $x, y \in P$ se cumple que $x \leq y$ implica $F(x) \leq' F(y)$. Escribiremos $F : (P, \leq) \to (P', \leq')$ para expresar que F es un homomorfismo de $(P, \leq)$ en $(P', \leq')$. Algunos ejemplos:

(E1) $F : \mathbf{R} \to \mathbf{R}$ dada por $F(r) = 3.r$ es un homomorfismo de $(\mathbf{R}, \leq)$ en $(\mathbf{R}, \leq)$

(E2) Sea $\leq\, = \{(n, m) \in \omega : n = m\}$ y sea $(P', \leq')$ un poset cualquiera. Entonces cualquier funcion $F : \omega \to P'$ es un homomorfismo de $(\omega, \leq)$ en $(P', \leq')$ (glup!)

(E3) Consideremos el poset $(\mathcal{P}(\omega), \leq)$, donde $\leq\, = \{(A, B) \in \mathcal{P}(\omega)^2 : A \subseteq B\}$ y el poset $(\mathcal{P}(\{1, 2, 3, 4\}), \leq')$, donde $\leq' = \{(A, B) \in \mathcal{P}(\{1, 2, 3, 4\})^2 : A \subseteq B\}$. Entonces $F : \mathcal{P}(\omega) \to \mathcal{P}(1, 2, 3, 4)$ dada por $F(A) = A \cap \{1, 2, 3, 4\}$ es un homomorfismo de $(\mathcal{P}(\omega), \leq)$ en $(\mathcal{P}(\{1, 2, 3, 4\}), \leq')$

Una funcion $F : P \to P'$ sera llamada un *isomorfismo de* $(P, \leq)$ *en* $(P', \leq')$ si F es biyectiva, F es un homomorfismo de $(P, \leq)$ en $(P', \leq')$ y F^{-1} es un homomorfismo de $(P', \leq')$ en $(P, \leq)$. Escribiremos $(P, \leq) \cong (P', \leq')$ cuando exista un isomorfismo de $(P, \leq)$ en $(P', \leq')$ y en tal caso diremos que $(P, \leq)$ y $(P', \leq')$ son *isomorfos*. Cabe observar que un homomorfismo biyectivo no necesariamente es un isomorfismo como lo muestra el siguiente ejemplo.

- Consideremos los posets $\mathbf{P} = (\{1, 2\}, \{(1, 1), (2, 2)\})$ y $\mathbf{Q} = (\{1, 2\}, \{(1, 1), (2, 2), (1, 2)\})$. Es facil ver que $F : \{1, 2\} \to \{1, 2\}$, dada por $F(1) = 1$ y $F(2) = 2$ es un homomorfismo de $\mathbf{P}$ en $\mathbf{Q}$. Dejamos al lector chequear que F^{-1} no es un homomorfismo de $\mathbf{Q}$ en $\mathbf{P}$.

Notacion: Dada una funcion $F : A \to B$ y $S \subseteq A$, denotaremos con $F(S)$ al conjunto $\{F(a) : a \in S\}$

El siguiente lema aporta evidencia al hecho de que los isomorfismos preservan todas las propiedades matematicas.

Lemma 139 *Sean $(P, \leq)$ y $(P', \leq')$ posets. Supongamos F es un isomorfismo de $(P, \leq)$ en $(P', \leq')$.*

(a) Para $x, y \in P$, tenemos que $x < y$ si y solo si $F(x) <' F(y)$.

(b) Para cada $x \in P$, se tiene que x es maximo (resp. minimo) de $(P, \leq)$ si y solo si $F(x)$ es maximo (resp. minimo) de $(P', \leq')$.

(c) Para cada $x \in P$, se tiene que x es maximal (resp. minimal) en $(P, \leq)$ si y solo si $F(x)$ es maximal (resp. minimal) en $(P', \leq')$.

(d) Para $x, y \in P$, tenemos que $x \prec y$ si y solo si $F(x) \prec' F(y)$.

(e) Para cada $S \subseteq P$ y cada $a \in P$, se tiene que a es cota superior (resp. inferior) de S si y solo si $F(a)$ es cota superior (resp. inferior) de $F(S)$.

(f) *Para cada $S \subseteq P$, se tiene que existe* $\sup(S)$ *si y solo si existe* $\sup(F(S))$ *y en el caso de que existan tales elementos se tiene que* $F(\sup(S)) = \sup(F(S))$.

(g) *Para* $x, y, z \in P$, *tenemos que* $z = \sup\{x, y\}$ *si y solo si* $F(z) = \sup\{F(x), F(y)\}$

(h) *Para* $x, y, z \in P$, *tenemos que* $z = \inf\{x, y\}$ *si y solo si* $F(z) = \inf\{F(x), F(y)\}$

Proof. (b) Sea $a \in P$ un elemento fijo. Supongamos $a \in P$ es maximo de $(P, \leq)$. Probaremos que $F(a)$ es maximo de $(P', \leq')$. Sea b un elemento fijo pero arbitrario de P'. Probaremos que $b \leq' F(a)$. Sea $d \in P$ tal que $F(d) = b$ (tal d existe y que F es sobreyectiva). Ya que a es maximo de $(P, \leq)$ tenemos que $d \leq a$. Ya que F es un homomorfismo tenemos que $F(d) \leq' F(a)$ por lo cual $b \leq' F(a)$ ya que $F(d) = b$. Ya que b era arbitrario hemos probado que $x \leq' F(a)$ para cada $x \in P'$, lo cual por definicion nos dice que $F(a)$ es maximo de $(P', \leq')$. Supongamos ahora que $F(a)$ es maximo de $(P', \leq')$. Probaremos que a es maximo de $(P, \leq)$. Sea b un elemento fijo pero arbitrario de P. Probaremos que $b \leq a$. Ya que $F(a)$ es maximo de $(P', \leq')$ tenemos que $F(b) \leq' F(a)$. Ya que F^{-1} es un homomorfismo tenemos que $F^{-1}(F(b)) \leq F^{-1}(F(a))$, por lo cual $b \leq a$. Ya que b era arbitrario hemos probado que $x \leq a$ para cada $x \in P$, lo cual por definicion nos dice que a es maximo de $(P, \leq)$.

Ya que a era fijo pero arbitrario hemos probado que cualquiera sea $x \in P$, se tiene que x es maximo de $(P, \leq)$ si y solo si $F(x)$ es maximo de $(P', \leq')$.

(e) Supongamos que a es cota superior de S. Veamos que entonces $F(a)$ es cota superior de $F(S)$. Sea $x \in F(S)$. Sea $s \in S$ tal que $x = F(s)$. Ya que $s \leq a$, tenemos que $x = F(s) \leq' F(a)$. Supongamos ahora que $F(a)$ es cota superior de $F(S)$ y veamos que entonces a es cota superior de S. Sea $s \in S$. Ya que $F(s) \leq' F(a)$, tenemos que $s = F^{-1}(F(s)) \leq F^{-1}(F(a)) = a$.

(f) Supongamos existe $\sup(S)$. Veamos entonces que $F(\sup(S))$ es el supremo de $F(S)$. Por (e) $F(\sup(S))$ es cota superior de $F(S)$. Supongamos b es cota superior de $F(S)$. Entonces $F^{-1}(b)$ es cota superior de S, por lo cual $\sup(S) \leq F^{-1}(b)$, produciendo $F(\sup(S)) \leq' b$. En forma analoga se ve que si existe $\sup(F(S))$, entonces $F^{-1}(\sup(F(S)))$ es el supremo de S.

Las pruebas faltantes son dejadas como ejercicio. ■

Notese que (d) nos garantiza que si dos posets finitos son isomorfos, entonces pueden representarse con el mismo diagrama de Hasse.

En la prueba de (b) del lema anterior se uso tacitamente la siguiente propiedad que es obvia pero clave en la demostracion:

- Si F es una funcion y $b \in \mathrm{Im}(F)$, entonces $b = F(d)$, para algun $d \in D_F$

Esto da lugar a la siguiente regla la cual es muy util a la hora de hacer pruebas:

Regla Pertenecer a la Imagen

Si ud en el desarrollo de una prueba conoce que un elemento b esta en la imagen de una funcion F, entonces escriba al elemento b en la forma $F(a)$, donde a denota algun elemento fijo del dominio de F

Muchas veces tener presente dicha regla es la diferencia a que a uno le salga o no una prueba determinada.

4.2 Version geometrica del concepto de reticulado

El concepto de reticulado puede ser abordado en dos formas distintas, una geometrica (via posets) y la otra algebraica (via estructuras algebraicas definidas ecuacionalmente). Como veremos mas adelante ambas definiciones son equivalentes.

Diremos que un poset $(P, \leq)$ es un *reticulado* si para todo $a, b \in P$, existen (en $(P, \leq)$) $\sup(\{a, b\})$ e $\inf(\{a, b\})$. algunos ejemplos:

(E1) El poset $(\mathbf{N}, D)$ es un reticulado ($D = \{(x, y) \in \mathbf{N}^2 : x|y\}$) ya que dados $x, y \in \mathbf{N}$, hemos visto que $mcd(x, y)$ y $mcm(x, y)$ son infimo y supremo del conjunto $\{x, y\}$ en $(\mathbf{N}, D)$

(E2) El poset $(\mathcal{P}(\omega), \leq)$ es un reticulado ($\leq\, = \{(A, B) \in \mathcal{P}(\omega)^2 : A \subseteq B\}$) ya que dados $A, B \in \mathcal{P}(\omega)$, hemos visto que $A \cap B$ y $A \cup B$ son infimo y supremo del conjunto $\{A, B\}$ en $(\mathcal{P}(\omega), \leq)$

Recordemos que dado un conjunto A, por una *operacion binaria sobre* A entenderemos una funcion cuyo dominio es A^2 y cuya imagen esta contenida en A. En un reticulado $(P, \leq)$ tenemos dos operaciones binarias naturalmente definidas:

$$\begin{array}{rcl} \mathsf{s} : P^2 & \rightarrow & P \\ (a, b) & \rightarrow & \sup(\{a, b\}) \end{array} \qquad \begin{array}{rcl} \mathsf{i} : P^2 & \rightarrow & P \\ (a, b) & \rightarrow & \inf(\{a, b\}) \end{array}$$

Escribiremos $a \mathrel{\mathsf{s}} b$ en lugar de $\mathsf{s}(a, b)$ y $a \mathrel{\mathsf{i}} b$ en lugar de $\mathsf{i}(a, b)$.

A continuacion nos dedicaremos a probar varias propiedades agradables que se cumplen en un reticulado $(L, \leq)$. Recomendamos al lector que en algunos casos practique encontrando pruebas perfectas. Esto lo entrenara en su capacidad de ser un matematico maduro, la cual sera crucial a la hora de hacer logica ya que la logica estudia (modeliza) matematicamente el funcionar de un matematico y sera muy practico que cada uno cuente con un matematico maduro en su propia mente

Lemma 140 *Dado un reticulado $(L, \leq)$ se cumplen las siguientes.*

(1) $x \leq x \mathrel{\mathsf{s}} y$, cualesquiera sean $x, y \in L$

(2) $x \mathsf{i} y \leq x$*, cualesquiera sean* $x, y \in L$

(3) $x \mathsf{s} x = x$*, cualesquiera sean* $x \in L$

(4) $x \mathsf{i} x = x$*, cualesquiera sean* $x \in L$

(5) $x \mathsf{s} y = y \mathsf{s} x$*, cualesquiera sean* $x, y \in L$

(6) $x \mathsf{i} y = y \mathsf{i} x$*, cualesquiera sean* $x, y \in L$

Proof. Prueba perfecta de (1). Sean $a, b \in L$, fijos. Por definicion de s tenemos que $a \mathsf{s} b = \sup(\{a, b\})$. O sea que por definicion de supremo de un conjunto tenemos que $a \mathsf{s} b$ es cota superior del conjunto $\{a, b\}$ en $(L \leq)$. O sea que $a \leq a \mathsf{s} b$. Ya que a, b eran arbitrarios, hemos probado que vale (1).

Prueba perfecta de (3). Sea $a \in L$, fijo. Por definicion de s tenemos que $a \mathsf{s} a = \sup(\{a, a\}) = \sup(\{a\})$. O sea que debemos probar que $a = \sup(\{a\})$. Es claro que a es cota superior de $\{a\}$. Ademas es claro que si z es cota superior de $\{a\}$, entonces $z \geq a$. O sea que por definicion de supremo de un conjunto tenemos que $a = \sup(\{a\})$. O sea que hemos probado que $a \mathsf{s} a = a$. Ya que a era arbitrario, hemos probado que vale (3).

Dejamos al lector completar la prueba. ■

Lemma 141 *Dado un reticulado* $(L, \leq)$ *se tiene que:*

(1) $x \leq y$ *si y solo si* $x \mathsf{s} y = y$*, cualesquiera sean* $x, y \in L$

(2) $x \leq y$ *si y solo si* $x \mathsf{i} y = x$*, cualesquiera sean* $x, y \in L$

Proof. Ejercicio ■

Las siguientes dos propiedades son conocidas como leyes de absorcion (por que?)

Lemma 142 *Dado un reticulado* $(L, \leq)$*, se tiene que:*

(1) $x \mathsf{s} (x \mathsf{i} y) = x$*, cualesquiera sean* $x, y \in L$

(2) $x \mathsf{i} (x \mathsf{s} y) = x$*, cualesquiera sean* $x, y \in L$

Proof. (1). Sean $a, b \in L$, fijos. Por definicion de i debemos probar que $\sup(\{a, a \mathsf{i} b\}) = a$. O sea debemos probar que a es la menor cota superior de $\{a, a \mathsf{i} b\}$. Por un lema anterior tenemos que $a \mathsf{i} b \leq a$ y obviamente se da $a \leq a$, lo cual nos dice que a es cota superior de $\{a, a \mathsf{i} b\}$. Es claro que es menor o igual que cualquier otra cota superior. O sea que hemos probado que $a \mathsf{s} (a \mathsf{i} b) = a$, lo cual ya que a, b eran elementos arbitrarios nos dice que vale (1).

(2) es dejada al lector. ■

Antes de seguir dando propiedades basicas de los reticulados daremos tres reglas que seran de suma utilidad para encontrar pruebas. Dejamos al lector justificar matematicamente la validez de dichas reglas.

Regla Igualdad en Posets

Si ud esta intentando probar que en un poset $(P, \leq)$ dos elementos x, y son iguales, desdoble su tarea en las dos tareas siguientes:

- Probar que $x \leq y$
- Probar que $y \leq x$

Regla Superar un Supremo

Si ud esta intentando probar que en un reticulado $(L, \leq)$ se da que $z \geq x \mathsf{s} y$, desdoble su tarea en las dos tareas siguientes:

- Probar que $z \geq x$
- Probar que $z \geq y$

Regla Ser Menor o Igual que un Infimo

Si ud esta intentando probar que en un reticulado $(L, \leq)$ se da que $z \leq x \mathsf{i} y$, desdoble su tarea en las dos tareas siguientes:

- Probar que $z \leq x$
- Probar que $z \leq y$

Ambas operaciones s e i son asociativas, es decir:

Lemma 143 *Dado un reticulado $(L, \leq)$, se tiene que:*

(1) $(x \mathsf{s} y) \mathsf{s} z = x \mathsf{s} (y \mathsf{s} z)$, cualesquiera sean $x, y, z \in L$

(2) $(x \mathsf{i} y) \mathsf{i} z = x \mathsf{i} (y \mathsf{i} z)$, cualesquiera sean $x, y, z \in L$

Proof. (1) Sean $a, b, c \in L$, fijos. Usaremos la regla Igualdad en Posets. Primero probaremos $(a \mathsf{s} b) \mathsf{s} c \leq a \mathsf{s} (b \mathsf{s} c)$. Para esto usaremos la regla Superar un Supremo. Es decir que debemos probar que

$$\begin{array}{rcl} (a \mathsf{s} b) & \leq & a \mathsf{s} (b \mathsf{s} c) \\ c & \leq & a \mathsf{s} (b \mathsf{s} c) \end{array}$$

Para la primer desigualdad usaremos tambien la regla Superar un Supremo, por lo cual deberemos probar

$$\begin{array}{rcl} a & \leq & a \mathsf{s} (b \mathsf{s} c) \\ b & \leq & a \mathsf{s} (b \mathsf{s} c) \end{array}$$

O sea que en suma debemos probar las siguientes desigualdades

$$\begin{array}{rcl} a & \leq & a \mathsf{s} (b \mathsf{s} c) \\ b & \leq & a \mathsf{s} (b \mathsf{s} c) \\ c & \leq & a \mathsf{s} (b \mathsf{s} c) \end{array}$$

La primera es directa de un lema anterior, y para la segunda notese que el mismo lema nos dice que

$$b \leq (b \mathsf{s} c) \text{ y } (b \mathsf{s} c) \leq a \mathsf{s} (b \mathsf{s} c)$$

por lo cual solo resta usar que $\leq$ es transitiva. La tercera es completamente analoga a la segunda.

En forma similar se prueba que $a \mathsf{s} (b \mathsf{s} c) \leq (a \mathsf{s} b) \mathsf{s} c$. Es decir que por la regla Igualdad en Posets tenemos que $a \mathsf{s} (b \mathsf{s} c) = (a \mathsf{s} b) \mathsf{s} c$. Ya que a, b, c eran elementos arbitrarios hemos probado que vale (1).

(2) es dejada como ejercicio. ■

El siguiente lema prueba que en un reticulado las operaciones s e i preservan el orden.

Lemma 144 *Dado un reticulado $(L, \leq)$, se tiene que:*

(1) $x \leq z$ e $y \leq w$ implica $x \mathsf{s} y \leq z \mathsf{s} w$, cualesquiera sean $x, y, z, w \in L$

(2) $x \leq z$ e $y \leq w$ implica $x \mathsf{i} y \leq z \mathsf{i} w$, cualesquiera sean $x, y, z, w \in L$

Proof. (1) Sean $a, b, c, d \in L$, elementos fijos. Supongamos que $a \leq c$ e $b \leq d$. Probaremos que entonces $a \mathsf{s} b \leq c \mathsf{s} d$. Por la regla Superar un Supremo basta con probar que

$$\begin{array}{rcl} a & \leq & c \mathsf{s} d \\ b & \leq & c \mathsf{s} d \end{array}$$

Para ver que $a \leq c \mathsf{s} d$ notese que $a \leq c$ (por hipotesis) y $c \leq c \mathsf{s} d$, por lo cual podemos usar que $\leq$ es transitiva. La desigualdad $b \leq c \mathsf{s} d$ se prueba en forma similar. O sea que hemos probado que

$$a \leq c \text{ y } b \leq d \text{ implica } a \mathsf{s} b \leq c \mathsf{s} d$$

Ya que a, b, c, d eran elementos arbitrarios hemos probado que vale (1).

(2) es dejada al lector ■

Lemma 145 *Dado un reticulado $(L, \leq)$, se tiene que:*

- *$(x \mathsf{i} y) \mathsf{s} (x \mathsf{i} z) \leq x \mathsf{i} (y \mathsf{s} z)$, cualesquiera sean $x, y, z \in L$*

Proof. Sean $a, b, c \in L$, elementos fijos. Por la regla Superar un Supremo, basta con probar que

$$\begin{array}{rcl} a \mathsf{~i~} b & \leq & a \mathsf{~i~} (b \mathsf{~s~} c) \\ a \mathsf{~i~} c & \leq & a \mathsf{~i~} (b \mathsf{~s~} c) \end{array}$$

Pero estas dos desigualdades pueden ser facilmente probadas aplicando (2) del lema anterior. O sea que $(a \mathsf{~i~} b) \mathsf{~s~} (a \mathsf{~i~} c) \leq a \mathsf{~i~} (b \mathsf{~s~} c)$, de lo cual se deduce nuestro lema ya que a, b, c eran elementos arbitrarios. ■

Iterar supremos (resp. infimos) da supremos (resp. infimos), es decir:

Lemma 146 *Sea $(L, \leq)$ un reticulado. Se tiene que*

$$\begin{array}{rcl} (...(x_1 \mathsf{~s~} x_2) \mathsf{~s~} ...) \mathsf{~s~} x_n & = & \sup(\{x_1, ..., x_n\}) \\ (...(x_1 \mathsf{~i~} x_2) \mathsf{~i~} ...) \mathsf{~i~} x_n & = & \inf(\{x_1, ..., x_n\}) \end{array}$$

cualesquiera sean los elementos $x_1, ..., x_n \in L$, con $n \geq 2$.

Proof. Por induccion en n. Claramente el resultado vale para $n = 2$. Supongamos vale para n y veamos entonces que vale para $n + 1$. Sean $a_1, ..., a_{n+1} \in L$, fijos. Por hipotesis inductiva tenemos que

(1) $(...(a_1 \mathsf{~s~} a_2) \mathsf{~s~} ...) \mathsf{~s~} a_n = \sup(\{a_1, ..., a_n\})$

Veamos entonces que

(2) $((...(a_1 \mathsf{~s~} a_2) \mathsf{~s~} ...) \mathsf{~s~} a_n) \mathsf{~s~} a_{n+1} = \sup(\{a_1, ..., a_{n+1}\})$

Usando (1), es facil ver que $((...(a_1 \mathsf{~s~} a_2) \mathsf{~s~} ...) \mathsf{~s~} a_n) \mathsf{~s~} a_{n+1}$ es cota superior de $\{a_1, ..., a_{n+1}\}$. Supongamos que z es otra cota superior. Ya que z es tambien cota superior del conjunto $\{a_1, ..., a_n\}$, por (1) tenemos que

$$(...(a_1 \mathsf{~s~} a_2) \mathsf{~s~} ...) \mathsf{~s~} a_n \leq z$$

Pero entonces ya que $a_{n+1} \leq z$, tenemos que

$$((...(a_1 \mathsf{~s~} a_2) \mathsf{~s~} ...) \mathsf{~s~} a_n) \mathsf{~s~} a_{n+1} \leq z$$

con lo cual hemos probado (2). Ya que $a_1, ..., a_{n+1} \in L$ eran elementos arbitrarios, hemos probado que vale el enunciado del lema para $n + 1$. ■

Dado que la distribucion de parentesis en una expresion de la forma

$$(...(x_1 \mathsf{~s~} x_2) \mathsf{~s~} ...) \mathsf{~s~} x_n$$

es irrelevante (ya que s es asociativa), en general suprimiremos los parentesis.

Concluimos esta subseccion enunciando una regla que hemos usado constantemente:

Regla Igualar un Supremo

1. Si ud esta intentando probar que en un poset $(P, \leq)$ se da que $x = \sup(S)$, desdoble su tarea en las dos tareas siguientes:

   - Probar que x es cota superior de S
   - Probar que si z es una cota superior de S, entonces $x \leq z$

Las cinco reglas consideradas estan muy vinculadas al concepto de inteligencia artificial ya que si quisieramos hacer un probador automatico del tipo de teoremas hechos en esta subseccion, claramente estas reglas le darian una alternativa de busqueda que podria (y de hecho lo hace) dar el camino adecuado para obtener la prueba de un enunciado dado.

4.3 Version algebraica del concepto de reticulado

De la diversas propiedades de las operaciones s e i de un reticulado $(L, \leq)$ distinguiremos las siguientes:

(I1) $x \mathsf{\ s\ } x = x \mathsf{\ i\ } x = x$, cualesquiera sea $x \in L$

(I2) $x \mathsf{\ s\ } y = y \mathsf{\ s\ } x$, cualesquiera sean $x, y \in L$

(I3) $x \mathsf{\ i\ } y = y \mathsf{\ i\ } x$, cualesquiera sean $x, y \in L$

(I4) $(x \mathsf{\ s\ } y) \mathsf{\ s\ } z = x \mathsf{\ s\ } (y \mathsf{\ s\ } z)$, cualesquiera sean $x, y, z \in L$

(I5) $(x \mathsf{\ i\ } y) \mathsf{\ i\ } z = x \mathsf{\ i\ } (y \mathsf{\ i\ } z)$, cualesquiera sean $x, y, z \in L$

(I6) $x \mathsf{\ s\ } (x \mathsf{\ i\ } y) = x$, cualesquiera sean $x, y \in L$

(I7) $x \mathsf{\ i\ } (x \mathsf{\ s\ } y) = x$, cualesquiera sean $x, y \in L$

Podemos abstraernos y pensar que s e i son dos operaciones cualesquiera sobre un conjunto L arbitrario y estudiar cuando se satisfacen y cuando no dichas propiedades. Por ejemplo si tomamos $L = \mathbf{R}$ y

$$\begin{array}{rcl} \mathsf{s} : \mathbf{R}^2 & \to & \mathbf{R} \\ (a, b) & \to & a + b \end{array} \qquad \begin{array}{rcl} \mathsf{i} : \mathbf{R}^2 & \to & \mathbf{R} \\ (a, b) & \to & a.b \end{array}$$

entonces se cumplen (I2), (I3), (I4) e (I5), pero (I1), (I6) e (I7) no se cumplen. Otro ejemplo, si tomamos $L = \{1, 2\}$ y

$$\begin{array}{rcl} \mathsf{s} : \{1,2\}^2 & \to & \{1,2\} \\ (1,1) & \to & 1 \\ (1,2) & \to & 2 \\ (2,1) & \to & 1 \\ (2,2) & \to & 2 \end{array} \qquad \begin{array}{rcl} \mathsf{i} : \{1,2\}^2 & \to & \{1,2\} \\ (1,1) & \to & 1 \\ (1,2) & \to & 1 \\ (2,1) & \to & 1 \\ (2,2) & \to & 1 \end{array}$$

entonces se cumplen (I3), (I4) e (I5), pero (I1), (I2), (I6) e (I7) no se cumplen. Un tercer ejemplo, si tomamos $L = \mathbf{N}$ y

$$\begin{array}{rcl} \mathsf{s} : \mathbf{N}^2 & \to & \mathbf{N} \\ (a,b) & \to & \max\{a,b\} \end{array} \qquad \begin{array}{rcl} \mathsf{i} : \mathbf{N}^2 & \to & \mathbf{N} \\ (a,b) & \to & \text{maximo comun divisor de } a \text{ y } b \end{array}$$

entonces se cumplen (I1), (I2), (I3), (I4), (I5) e (I6), pero (I7) no se cumple. Por supuesto si s e i son las operaciones supremo e infimo dadas por algun orden parcial $\leq$ sobre L el cual hace de $(L, \leq)$ un reticulado, entonces las propedades (I1),...,(I7) se cumplen y esto es justamente lo probado en la ultima serie de lemas. El ultimo ejemplo nos permite ver una sutileza. Notese que en este ejemplo s es la operacion supremo del reticulado $(\mathbf{N}, \leq)$, donde $\leq$ es el orden usual de los naturales, e i es la operacion infimo del reticulado $(\mathbf{N}, |)$, donde $|$ es el orden de la divisibilidad de los naturales. Sin envargo la ultima propiedad falla y esto se debe a que s e i son supremo e infimo pero respecto de distintos ordenes parciales.

Lo anterior motiva la siguiente definicion: Una terna $(L, \mathsf{s}, \mathsf{i})$, donde L es un conjunto no vacio y s e i son dos operaciones binarias sobre L sera llamada *reticulado* cuando cumpla (I1),...,(I7). Si $(L, \mathsf{s}, \mathsf{i})$ es un reticulado, llamaremos a L el *universo* de $(L, \mathsf{s}, \mathsf{i})$.

Observacion importante: Notese que hemos llamado a ciertos posets reticulados y a ciertas ternas reticulados pero deberia quedar claro que nunca un poset que es un reticulado puede ser igual a una terna que es un reticulado ya que son objetos matematicos de distinta naturaleza.

Tal como lo vimos recien, las ternas dadas por los tres ejemplos anteriores no son reticulados ya que fallan alguna de las identidades (I1),...,(I7), y si tomamos un poset $(L, \leq)$ el cual sea un reticulado, entonces la terna $(L, \mathsf{s}, \mathsf{i})$, con s e i definidas como supremo e infimo, es un reticulado. El siguiente teorema muestra que todo reticulado $(L, \mathsf{s}, \mathsf{i})$ se obtiene de esta forma.

Theorem 147 (Dedekind) *Sea $(L, \mathsf{s}, \mathsf{i})$ un reticulado. La relacion binaria definida por:*

$$x \leq y \text{ si y solo si } x \,\mathsf{s}\, y = y$$

es un orden parcial sobre L para el cual se cumple que:

$$\begin{array}{rcl} \sup(\{x, y\}) & = & x \,\mathsf{s}\, y \\ \inf(\{x, y\}) & = & x \,\mathsf{i}\, y \end{array}$$

cualesquiera sean $x, y \in L$

Proof. Dejamos como ejercicio para el lector probar que $\leq$ es reflexiva y antisimetrica con respecto a L. Veamos que $\leq$ es transitiva con respecto a L. Supongamos que $x \leq y$ e $y \leq z$. Es decir que por definicion de $\leq$ tenemos que

$$\begin{array}{rcl} x \,\mathsf{s}\, y & = & y \\ y \,\mathsf{s}\, z & = & z \end{array}$$

Entonces

$$x \,\mathsf{s}\, z = x \,\mathsf{s}\, (y \,\mathsf{s}\, z) = (x \,\mathsf{s}\, y) \,\mathsf{s}\, z = y \,\mathsf{s}\, z = z$$

por lo cual $x \leq z$. O sea que ya sabemos que $(L, \leq)$ es un poset. Veamos ahora que $\sup(\{x,y\}) = x \,\mathsf{s}\, y$. Primero debemos ver que $x \,\mathsf{s}\, y$ es una cota superior del conjunto $\{x,y\}$, es decir

$$\begin{array}{rcl} x & \leq & x \,\mathsf{s}\, y \\ y & \leq & x \,\mathsf{s}\, y \end{array}$$

Por la definicion de $\leq$ debemos probar que

$$\begin{array}{rcl} x \,\mathsf{s}\, (x \,\mathsf{s}\, y) & = & x \,\mathsf{s}\, y \\ y \,\mathsf{s}\, (x \,\mathsf{s}\, y) & = & x \,\mathsf{s}\, y \end{array}$$

Estas igualdades se pueden probar usando (I1), (I2) y (I4). Dejamos al lector hacerlo como ejercicio.

Nos falta ver entonces que $x \,\mathsf{s}\, y$ es menor o igual que cualquier cota superior de $\{x,y\}$. Supongamos $x, y \leq z$. Es decir que por definicion de $\leq$ tenemos que

$$\begin{array}{rcl} x \,\mathsf{s}\, z & = & z \\ y \,\mathsf{s}\, z & = & z \end{array}$$

Pero entonces

$$(x \,\mathsf{s}\, y) \,\mathsf{s}\, z = x \,\mathsf{s}\, (y \,\mathsf{s}\, z) = x \,\mathsf{s}\, z = z$$

por lo que $x \,\mathsf{s}\, y \leq z$. Es decir que $x \,\mathsf{s}\, y$ es la menor cota superior.

Para probar que $\inf(\{x,y\}) = x \,\mathsf{i}\, y$, probaremos que para todo $u, v \in L$,

$$u \leq v \text{ si y solo si } u \,\mathsf{i}\, v = u$$

lo cual le permitira al lector aplicar un razonamiento similar al usado en la prueba de que $\sup(\{x,y\}) = x \,\mathsf{s}\, y$. Supongamos que $u \leq v$. Por definicion tenemos que $u \,\mathsf{s}\, v = v$. Entonces

$$u \,\mathsf{i}\, v = u \,\mathsf{i}\, (u \,\mathsf{s}\, v)$$

Pero por (I7) tenemos que $u \,\mathsf{i}\, (u \,\mathsf{s}\, v) = u$, lo cual implica $u \,\mathsf{i}\, v = u$. Reciprocamente si $u \,\mathsf{i}\, v = u$, entonces

$$\begin{array}{rcl} u \,\mathsf{s}\, v & = & (u \,\mathsf{i}\, v) \,\mathsf{s}\, v \\ & = & v \,\mathsf{s}\, (u \,\mathsf{i}\, v) \text{ (por (I2))} \\ & = & v \,\mathsf{s}\, (v \,\mathsf{i}\, u) \text{ (por (I3))} \\ & = & v \text{ (por (I6))} \end{array}$$

lo cual nos dice que $u \leq v$. $\blacksquare$

Ejercicio: Use los resultados anteriores para definir una funcion $\mathcal{F}$ de $\{(L,\leq):(L,\leq)$ es un reticulado$\}$ en $\{(L,\mathsf{s},\mathsf{i}):(L,\mathsf{s},\mathsf{i})$ es un reticulado$\}$ la cual sea biyectiva

Refleccion Informatica

Como vimos recien a nivel de informacion es lo mismo tener un poset que es un reticulado que una terna que es un reticulado. Es decir, los dos conceptos pueden considerarse dos formas distintas de presentar la misma informacion. Muchas veces es mas facil especificar un reticulado dando el poset ya que simplemente podemos dar su diagrama de Hasse y esto en general es una forma economica de dar las operaciones s e i.

Recordemos que algo similar sucedia con los conceptos equivalentes de relacion de equivalencia y particion.

4.3.1 El orden asociado a un reticulado

Como vimos el teorema de Dedekind nos dice que un reticulado $(L,\mathsf{s},\mathsf{i})$ es un objeto geometrico ya que si definimos

$$\leq\,=\{(x,y):x\;\mathsf{s}\;y=y\}$$

entonces $\leq$ es un orden parcial sobre L y las operaciones s e i resultan ser supremo e infimo. Llamaremos a $\leq\,=\{(x,y):x\;\mathsf{s}\;y=y\}$ el *orden parcial asociado a* $(L,\mathsf{s},\mathsf{i})$ y a $(L,\leq)$ el *poset asociado a* $(L,\mathsf{s},\mathsf{i})$. Notese que tambien tenemos que $\leq\,=\{(x,y):x\;\mathsf{i}\;y=x\}$ (¿por que?). Muchos conceptos definidos para posets ahora pueden aplicarse cuando tenemos un reticulado $(L,\mathsf{s},\mathsf{i})$. Por ejemplo, si decimos que $(L,\mathsf{s},\mathsf{i})$ tiene elemento maximo, esto significara que el poset $(L,\leq)$ tiene elemento maximo. Otro ejemplo, si decimos que en $(L,\mathsf{s},\mathsf{i})$ se da que el supremo de un conjunto S es a, nos estaremos refiriendo a que en su poset asociado $(L,\leq)$ se da que el supremo de S es a.

4.3.2 Notacion

Usaremos las siguientes practicas convenciones notacionales

Convencion notacional 1: Si L es un conjunto no vacio cuyos elementos son conjuntos y L cumple la siguiente condicion

- Si $A,B\in L$, entonces $A\cup B, A\cap B\in L$

entonces ciertas veces usaremos $\cup$ (resp. $\cap$) para denotar la operacion binaria sobre L dada por la union (resp. la intercecion). Es decir $\cup$ e $\cap$ denotaran las funciones

$$\begin{array}{rcl} L^2 & \rightarrow & L \\ (A,B) & \rightarrow & A\cup B \end{array} \qquad\qquad \begin{array}{rcl} L^2 & \rightarrow & L \\ (A,B) & \rightarrow & A\cap B \end{array}$$

Convencion notacional 2: Si L es un conjunto no vacio cuyos elementos son numeros reales entonces ciertas veces usaremos max y min para denotar las operaciones binarias sobre L dadas por

$$\begin{array}{rcl} L^2 & \rightarrow & L \\ (a,b) & \rightarrow & \max(a,b) \end{array} \qquad \begin{array}{rcl} L^2 & \rightarrow & L \\ (a,b) & \rightarrow & \min(a,b) \end{array}$$

Convencion notacional 3: Si L es un conjunto no vacio cuyos elementos son numeros naturales y L cumple la siguiente condicion

- Si $a, b \in L$, entonces $mcm(a,b), mcd(a,b) \in L$

entonces ciertas veces usaremos mcm y mcd para denotar las operaciones binarias sobre L dadas por

$$\begin{array}{rcl} L^2 & \rightarrow & L \\ (a,b) & \rightarrow & mcm(a,b) \end{array} \qquad \begin{array}{rcl} L^2 & \rightarrow & L \\ (a,b) & \rightarrow & mcd(a,b) \end{array}$$

Convencion notacional 4: Si P es un conjunto no vacio contenido en $\mathbf{N}$, entonces escribiremos $(P, |)$ para denotar al poset $(P, \{(x,y) \in P^2 : x|y\})$. Similarmente si P es un conjunto cuyos elementos son conjuntos, entonces escribiremos $(P, \subseteq)$ para denotar al poset $(P, \{(A,B) \in P^2 : A \subseteq B\})$

En virtud de las convenciones notacionales anteriores notese que por ejemplo

1. $(\mathbf{R}, \max, \min)$
2. $([0,1], \max, \min)$
3. $(\mathcal{P}(\mathbf{N}), \cup, \cap)$
4. $(\{A \subseteq \mathbf{N} : A \text{ es finito}\}, \cup, \cap)$
5. $(\mathbf{N}, mcm, mcd)$
6. $(\{1,2,3,6,12\}, mcm, mcd)$

denotan reticulados pero deberia quedar claro que en los primeros dos ejemplos max denota dos distintas operaciones. Analogamente sucede con min, $\cup$, $\cap$, mcm y mcd.

Similarmente

1. $(\mathbf{N}, |)$
2. $(\{1,2,3,6,7\}, |)$
3. $(\{\{1\}, \{1,7\}, \{1,2,3\}, \{16,99,65\}\}, \subseteq)$
4. $(\{A \subseteq \mathbf{N} : A \text{ es finito}\}, \subseteq)$

denotan posets pero deberia quedar claro que en los primeros dos ejemplos $|$ denota dos distintos ordenes parciales. Analogamente sucede con $\subseteq$

Estas ambiguedades no nos traeran problemas si estamos atentos al contexto.

4.3.3 Subreticulados

Dados reticulados $(L, \mathsf{s}, \mathsf{i})$ y $(L', \mathsf{s}', \mathsf{i}')$ diremos que $(L, \mathsf{s}, \mathsf{i})$ *es un subreticulado de* $(L', \mathsf{s}', \mathsf{i}')$ si se dan las siguientes condiciones

(1) $L \subseteq L'$

(2) $\mathsf{s} = \mathsf{s}'|_{L \times L}$ y $\mathsf{i} = \mathsf{i}'|_{L \times L}$

Notese que la condicion (2) en particular implica que L es cerrado bajo las operaciones s' e i' (i.e. $x \mathsf{ s}' y$, $x \mathsf{ i}' y \in L$, para todo $x, y \in L$). Esto motiva la siguiente definicion:

Sea $(L, \mathsf{s}, \mathsf{i})$ un reticulado. Un conjunto $S \subseteq L$ es llamado *subuniverso* de $(L, \mathsf{s}, \mathsf{i})$ si es no vacio y cerrado bajo las operaciones s e i (i.e. $x \mathsf{ s } y$, $x \mathsf{ i } y \in S$, para todo $x, y \in S$). Es importante notar que si bien los conceptos de subreticulado y subuniverso estan muy relacionados, se trata de conceptos diferentes ya que los subreticulados de $(L, \mathsf{s}, \mathsf{i})$ son reticulados, es decir ternas y los subuniversos de $(L, \mathsf{s}, \mathsf{i})$ son conjuntos, por lo cual no son ternas.

Es facil de chequear que si S es un subuniverso de $(L, \mathsf{s}, \mathsf{i})$, entonces $(S, \mathsf{s}|_{S \times S}, \mathsf{i}|_{S \times S})$ es un subreticulado de $(L, \mathsf{s}, \mathsf{i})$ y que todo subreticulado de $(L, \mathsf{s}, \mathsf{i})$ se obtiene en esta forma. Es decir, hay una biyeccion entre el conjunto de los subreticulados de $(L, \mathsf{s}, \mathsf{i})$ y el conjunto de los subuniversos de $(L, \mathsf{s}, \mathsf{i})$ (cual es?). Dicho de manera mas rapida: los subuniversos de $(L, \mathsf{s}, \mathsf{i})$ son ni mas ni menos que los universos de los subreticulados de $(L, \mathsf{s}, \mathsf{i})$.

4.3.4 Homomorfismos de reticulados

Sean $(L, \mathsf{s}, \mathsf{i})$ y $(L', \mathsf{s}', \mathsf{i}')$ reticulados. Una funcion $F : L \to L'$ sera llamada un *homomorfismo de* $(L, \mathsf{s}, \mathsf{i})$ *en* $(L', \mathsf{s}', \mathsf{i}')$ si para todo $x, y \in L$ se cumple que

$$\begin{array}{rcl} F(x \mathsf{ s } y) & = & F(x) \mathsf{ s}' F(y) \\ F(x \mathsf{ i } y) & = & F(x) \mathsf{ i}' F(y). \end{array}$$

Un homomorfismo de $(L, \mathsf{s}, \mathsf{i})$ en $(L', \mathsf{s}', \mathsf{i}')$ sera llamado *isomorfismo de* $(L, \mathsf{s}, \mathsf{i})$ *en* $(L', \mathsf{s}', \mathsf{i}')$ cuando sea biyectivo y su inversa sea tambien un homomorfismo. Escribiremos $(L, \mathsf{s}, \mathsf{i}) \cong (L', \mathsf{s}', \mathsf{i}')$ cuando exista un isomorfismo de $(L, \mathsf{s}, \mathsf{i})$ en $(L', \mathsf{s}', \mathsf{i}')$. Escribiremos "Sea $F : (L, \mathsf{s}, \mathsf{i}) \to (L', \mathsf{s}', \mathsf{i}')$ un homomorfismo" para expresar que F es un homomorfismo de $(L, \mathsf{s}, \mathsf{i})$ en $(L', \mathsf{s}', \mathsf{i}')$. No hay que confundirse al leer esta notacion y pensar que F es una funcion cuyo dominio es $(L, \mathsf{s}, \mathsf{i})$, lo cual por otra parte no tiene sentido ya que el dominio de una funcion nunca puede ser una 3-upla!

Lemma 148 *Si* $F : (L, \mathsf{s}, \mathsf{i}) \to (L', \mathsf{s}', \mathsf{i}')$ *es un homomorfismo biyectivo, entonces* F *es un isomorfismo*

Proof. Solo falta ver que F^{-1} es un homomorfismo. Sean $F(x), F(y)$ dos elementos cualesquiera de L'. Tenemos que

$$F^{-1}(F(x) \mathsf{ s}' F(y)) = F^{-1}(F(x \mathsf{ s } y)) = x \mathsf{ s } y = F^{-1}(F(x)) \mathsf{ s } F^{-1}(F(y))$$

■

Lemma 149 *Sean* $(L, \mathsf{s}, \mathsf{i})$ *y* $(L', \mathsf{s}', \mathsf{i}')$ *reticulados y sea* $F : (L, \mathsf{s}, \mathsf{i}) \to (L', \mathsf{s}', \mathsf{i}')$ *un homomorfismo. Entonces* I_F *es un subuniverso de* $(L', \mathsf{s}', \mathsf{i}')$*. Es decir que* F *es tambien un homomorfismo de* $(L, \mathsf{s}, \mathsf{i})$ *en* $(I_F, \mathsf{s}'|_{I_F \times I_F}, \mathsf{i}'|_{I_F \times I_F})$

Proof. Ya que L es no vacio tenemos que I_F tambien es no vacio. Sean $a, b \in I_F$. Sean $x, y \in L$ tales que $F(x) = a$ y $F(y) = b$. Se tiene que

$$\begin{aligned} a \,\mathsf{s}'\, b &= F(x) \,\mathsf{s}'\, F(y) = F(x \,\mathsf{s}\, y) \in I_F \\ a \,\mathsf{i}'\, b &= F(x) \,\mathsf{i}'\, F(y) = F(x \,\mathsf{i}\, y) \in I_F \end{aligned}$$

por lo cual I_F es cerrada bajo s' e i'. ■

Lemma 150 *Sean* $(L, \mathsf{s}, \mathsf{i})$ *y* $(L', \mathsf{s}', \mathsf{i}')$ *reticulados y sean* $(L \leq)$ *y* $(L', \leq')$ *los posets asociados. Sea* $F : L \to L'$ *una funcion. Entonces* F *es un isomorfismo de* $(L, \mathsf{s}, \mathsf{i})$ *en* $(L', \mathsf{s}', \mathsf{i}')$ *si y solo si* F *es un isomorfismo de* $(L, \leq)$ *en* $(L', \leq')$.

Proof. Supongamos F es un isomorfismo de $(L, \mathsf{s}, \mathsf{i})$ en $(L', \mathsf{s}', \mathsf{i}')$. Sean $x, y \in L$, tales que $x \leq y$. Tenemos que $y = x \,\mathsf{s}\, y$ por lo cual $F(y) = F(x \,\mathsf{s}\, y) = F(x) \,\mathsf{s}'\, F(y)$, produciendo $F(x) \leq' F(y)$. En forma similar se puede ver que F^{-1} es tambien un homomorfismo de $(L', \leq')$ en $(L, \leq)$. Si F es un isomorfismo de $(L, \leq)$ en $(L', \leq')$, entonces (g) y (h) del Lema 139 nos dicen que F y F^{-1} son homomorfismos (de reticulados terna) por lo cual F es un isomorfismo de $(L, \mathsf{s}, \mathsf{i})$ en $(L', \mathsf{s}', \mathsf{i}')$. ■

Ejercicio: Encontrar dos reticulados, $(L, \mathsf{s}, \mathsf{i})$ y $(L', \mathsf{s}', \mathsf{i}')$, tales que haya una función biyectiva de L en L' que preserve orden pero no sea homomorfismo de reticulados.

4.3.5 Congruencias de reticulados

Sea $(L, \mathsf{s}, \mathsf{i})$ un reticulado. Una *congruencia sobre* $(L, \mathsf{s}, \mathsf{i})$ sera una relacion de equivalencia θ sobre L la cual cumpla:

(1) $x\theta x'$ y $y\theta y'$ implica $(x \,\mathsf{s}\, y)\theta(x' \,\mathsf{s}\, y')$ y $(x \,\mathsf{i}\, y)\theta(x' \,\mathsf{i}\, y')$

Gracias a esta condicion podemos definir sobre L/θ dos operaciones binarias $\tilde{\mathsf{s}}$ e $\tilde{\imath}$, de la siguiente manera:

$$\begin{aligned} x/\theta \,\tilde{\mathsf{s}}\, y/\theta &= (x \,\mathsf{s}\, y)/\theta \\ x/\theta \,\tilde{\imath}\, y/\theta &= (x \,\mathsf{i}\, y)/\theta \end{aligned}$$

La terna $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath})$ es llamada el *cociente de* $(L, \mathsf{s}, \mathsf{i})$ *sobre* θ y la denotaremos con $(L, \mathsf{s}, \mathsf{i})/\theta$.

Lemma 151 *$(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath})$ es un reticulado. El orden parcial $\tilde{\leq}$ asociado a este reticulado cumple*

$$x/\theta \tilde{\leq} y/\theta \text{ sii } y\theta(x \mathsf{s} y)$$

Proof. Veamos que la estructura $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath})$ cumple (I4). Sean x/θ, y/θ, z/θ elementos cualesquiera de L/θ. Tenemos que

$$\begin{aligned} (x/\theta \,\tilde{\mathsf{s}}\, y/\theta) \,\tilde{\mathsf{s}}\, z/\theta &= (x \mathsf{s} y)/\theta \,\tilde{\mathsf{s}}\, z/\theta \\ &= ((x \mathsf{s} y) \mathsf{s} z)/\theta \\ &= (x \mathsf{s} (y \mathsf{s} z))/\theta \\ &= x/\theta \,\tilde{\mathsf{s}}\, (y \mathsf{s} z)/\theta \\ &= x/\theta \,\tilde{\mathsf{s}}\, (y/\theta \,\tilde{\mathsf{s}}\, z/\theta) \end{aligned}$$

En forma similar se puede ver que la estructura $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath})$ cumple el resto de las identidades que definen reticulado.

Por definicion, $x/\theta \tilde{\leq} y/\theta$ sii $y/\theta = x/\theta \,\tilde{\mathsf{s}}\, y/\theta$, por lo cual $x/\theta \tilde{\leq} y/\theta$ sii $y/\theta = (x \mathsf{s} y)/\theta$. ■

El reticulado $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath})$ es llamado el *cociente de* $(L, \mathsf{s}, \mathsf{i})$ *sobre* θ.

Corollary 152 *Sea $(L, \mathsf{s}, \mathsf{i})$ un reticulado en el cual hay un elemento maximo 1 (resp. minimo 0). Entonces si θ es una congruencia sobre $(L, \mathsf{s}, \mathsf{i})$, $1/\theta$ (resp. $0/\theta$) es un elemento maximo (resp. minimo) de $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath})$.*

Proof. Ya que $1 = x \mathsf{s} 1$, para cada $x \in L$, tenemos que $1/\theta = (x \mathsf{s} 1)/\theta$, para cada $x \in L$, lo cual por el lema anterior nos dice que $x/\theta \tilde{\leq} 1/\theta$, para cada $x \in L$. ■

El siguiente lema nos da una forma natural de encontrar congruencias

Lemma 153 *Si $F : (L, \mathsf{s}, \mathsf{i}) \to (L', \mathsf{s}', \mathsf{i}')$ es un homomorfismo, entonces $\ker F$ es una congruencia sobre $(L, \mathsf{s}, \mathsf{i})$.*

Proof. Dejamos al lector ver que $\ker F$ es una relacion de equivalencia. Supongamos $x \ker F x'$ y $y \ker F y'$. Entonces

$$F(x \mathsf{s} y) = F(x) \mathsf{s}' F(y) = F(x') \mathsf{s}' F(y') = F(x' \mathsf{s} y')$$

lo cual nos dice que $(x \mathsf{s} y) \ker F (x' \mathsf{s} y')$. En forma similar tenemos que $(x \mathsf{i} y) \ker F (x' \mathsf{i} y')$. ■

Ya vimos que el nucleo de un homomorfismo es una congruencia. El siguiente lema muestra que toda congruencia es el nucleo de un homomorfismo.

Lemma 154 *Sea $(L, \mathsf{s}, \mathsf{i})$ un reticulado y sea θ una congruencia sobre $(L, \mathsf{s}, \mathsf{i})$. Entonces π_θ es un homomorfismo de $(L, \mathsf{s}, \mathsf{i})$ en $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath})$. Ademas $\ker \pi_\theta = \theta$.*

Proof. Sean $x, y \in L$. Tenemos que

$$\pi_\theta(x \mathsf{s} y) = (x \mathsf{s} y)/\theta = x/\theta \,\tilde{\mathsf{s}}\, y/\theta = \pi_\theta(x) \,\tilde{\mathsf{s}}\, \pi_\theta(y)$$

por lo cual π_θ preserva la operacion supremo. Para la operacion infimo es similar. ■

4.4 Reticulados acotados

Por un *reticulado acotado* entenderemos una 5-upla $(L, \mathsf{s}, \mathsf{i}, 0, 1)$, tal que $(L, \mathsf{s}, \mathsf{i})$ es un reticulado, $0, 1 \in L$, y ademas se cumplen las siguientes identidades

(I8) $0 \mathbin{\mathsf{s}} x = x$, para cada $x \in L$

(I9) $x \mathbin{\mathsf{s}} 1 = 1$, para cada $x \in L$.

Por ejemplo $(\{4, 56, 449\}, \max, \min, 4, 449)$ es un reticulado acotado pero es facil ver que $(\{4, 56, 449\}, \max, \min, 449, 56)$ no lo es.

Reflexion Informatica

Por supuesto, en virtud de lo desarrollado en la subseccion anterior se tiene que si $(L, \leq)$ es un poset el cual es un reticulado y en el cual hay un maximo 1 y un minimo 0, entonces si tomamos:

$$\begin{array}{rcl} \mathsf{s} : L^2 & \rightarrow & L \\ (a,b) & \rightarrow & \sup(\{a,b\}) \end{array} \qquad\qquad \begin{array}{rcl} \mathsf{i} : L^2 & \rightarrow & L \\ (a,b) & \rightarrow & \inf(\{a,b\}) \end{array}$$

tenemos que $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ es un reticulado acotado. Ademas en virtud del Teorema de Dedekind todo reticulado acotado se obtiene de esta forma. O sea que a nivel de informacion, un poset que es un reticulado y tiene 0 y 1 es exactamente lo mismo que un reticulado acotado

4.4.1 Subreticulados acotados

Dados reticulados acotados $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ y $(L', \mathsf{s}', \mathsf{i}', 0', 1')$ diremos que $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ *es un subreticulado acotado de* $(L', \mathsf{s}', \mathsf{i}', 0', 1')$ si se dan las siguientes condiciones

(1) $L \subseteq L'$

(2) $0 = 0'$ y $1 = 1'$

(3) $\mathsf{s} = \mathsf{s}'|_{L \times L}$ y $\mathsf{i} = \mathsf{i}'|_{L \times L}$

Tal como para el caso de reticulados como ternas tenemos el concepto de subuniverso el cual definimos a continuacion:

Sea $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ un reticulado acotado. Un conjunto $S \subseteq L$ es llamado *subuniverso* de $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ si $0, 1 \in S$ y ademas S es cerrado bajo las operaciones s e i (i.e. $x \mathbin{\mathsf{s}} y$, $x \mathbin{\mathsf{i}} y \in S$, para todo $x, y \in S$). Es importante notar que si bien los conceptos de subreticulado acotado y subuniverso estan muy relacionados, se trata de conceptos diferentes ya que los subreticulados acotados de $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ son reticulados acotados, es decir 5-uplas y los subuniversos de $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ son conjuntos, por lo cual no son 5-uplas.

Es facil de chequear que si S es un subuniverso de $(L, \mathsf{s}, \mathsf{i}, 0, 1)$, entonces $(S, \mathsf{s}|_{S \times S}, \mathsf{i}|_{S \times S}, 0, 1)$ es un subreticulado acotado de $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ y que todo subreticulado acotado de $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ se obtiene en esta forma. Es decir, hay una biyeccion entre el conjunto de los subreticulados acotados de $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ y el conjunto de los subuniversos de $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ (cual es?). Dicho de manera mas rapida: los subuniversos de $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ son ni mas ni menos que los universos de los subreticulados acotados de $(L, \mathsf{s}, \mathsf{i}, 0, 1)$.

4.4.2 Homomorfismos de reticulados acotados

Sean $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ y $(L', \mathsf{s}', \mathsf{i}', 0', 1')$ reticulados acotados. Una funcion $F : L \rightarrow L'$ sera llamada un *homomorfismo de* $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ *en* $(L', \mathsf{s}', \mathsf{i}', 0', 1')$ si para todo $x, y \in L$ se cumple que

$$\begin{array}{rcl} F(x \mathsf{~s~} y) & = & F(x) \mathsf{~s}' F(y) \\ F(x \mathsf{~i~} y) & = & F(x) \mathsf{~i}' F(y) \\ F(0) & = & 0' \\ F(1) & = & 1' \end{array}$$

Un homomorfismo de $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ en $(L', \mathsf{s}', \mathsf{i}', 0', 1')$ sera llamado *isomorfismo* cuando sea biyectivo y su inversa sea tambien un homomorfismo. Escribiremos $(L, \mathsf{s}, \mathsf{i}, 0, 1) \cong (L', \mathsf{s}', \mathsf{i}', 0', 1')$ cuando exista un isomorfismo de $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ en $(L', \mathsf{s}', \mathsf{i}', 0', 1')$. Escribiremos "Sea $F : (L, \mathsf{s}, \mathsf{i}, 0, 1) \rightarrow (L', \mathsf{s}', \mathsf{i}', 0', 1')$ un homomorfismo" para expresar que F es un homomorfismo de $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ en $(L', \mathsf{s}', \mathsf{i}', 0', 1')$. No hay que confundirse al leer esta notacion y pensar que F es una funcion cuyo dominio es $(L, \mathsf{s}, \mathsf{i}, 0, 1)$, lo cual por otra parte no tiene sentido ya que el dominio de una funcion nunca puede ser una 5-upla!

Lemma 155 *Si* $F : (L, \mathsf{s}, \mathsf{i}, 0, 1) \rightarrow (L', \mathsf{s}', \mathsf{i}', 0', 1')$ *un homomorfismo biyectivo, entonces* F *es un isomorfismo*

Proof. Similar a la prueba del Lemma 148. ■

Lemma 156 *Si* $F : (L, \mathsf{s}, \mathsf{i}, 0, 1) \rightarrow (L', \mathsf{s}', \mathsf{i}', 0', 1')$ *es un homomorfismo, entonces* I_F *es un subuniverso de* $(L', \mathsf{s}', \mathsf{i}', 0', 1')$. *Es decir que* F *es tambien un homomorfismo de* $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ *en* $(I_F, \mathsf{s}'|_{I_F \times I_F}, \mathsf{i}'|_{I_F \times I_F}, 0', 1')$

Proof. Ya que F es un homomorfismo de $(L, \mathsf{s}, \mathsf{i})$ en $(L', \mathsf{s}', \mathsf{i}')$ tenemos que I_F es subuniverso de $(L', \mathsf{s}', \mathsf{i}')$ lo cual ya que $0', 1' \in I_F$ implica que I_F es un subuniverso de $(L', \mathsf{s}', \mathsf{i}', 0', 1')$. ■

4.4.3 Congruencias de reticulados acotados

Sea $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ un reticulado acotado. Una *congruencia sobre* $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ sera una relacion de equivalencia θ la cual sea una congruencia sobre $(L, \mathsf{s}, \mathsf{i})$. Tenemos definidas sobre L/θ dos operaciones binarias $\tilde{\mathsf{s}}$ e $\tilde{\imath}$, de la siguiente manera:

$$\begin{array}{rcl} x/\theta \tilde{\mathsf{s}} y/\theta & = & (x \mathsf{~s~} y)/\theta \\ x/\theta \tilde{\imath} y/\theta & = & (x \mathsf{~i~} y)/\theta \end{array}$$

La 5-upla $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath}, 0/\theta, 1/\theta)$ es llamada el *cociente de* $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ *sobre* θ y la denotaremos con $(L, \mathsf{s}, \mathsf{i}, 0, 1)/\theta$.

Lemma 157 *Sea* $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ *un reticulado acotado y* θ *una congruencia sobre* $(L, \mathsf{s}, \mathsf{i}, 0, 1)$.

(a) $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath}, 0/\theta, 1/\theta)$ *es un reticulado acotado.*

(b) π_θ *es un homomorfismo de* $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ *en* $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath}, 0/\theta, 1/\theta)$ *cuyo nucleo es* θ.

Proof. (a) Es facil ver que $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath}, 0/\theta, 1/\theta)$ cumple (I1), (I2),...,(I9) dado que $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ las cumple.

(b) Sigue directamente del Lema 154 ■

Lemma 158 *Si* $F : (L, \mathsf{s}, \mathsf{i}, 0, 1) \to (L', \mathsf{s}', \mathsf{i}', 0', 1')$ *es un homomorfismo de reticulados acotados, entonces* $\ker F$ *es una congruencia sobre* $(L, \mathsf{s}, \mathsf{i}, 0, 1)$.

Proof. Ya que F es un homomorfismo de $(L, \mathsf{s}, \mathsf{i})$ en $(L', \mathsf{s}', \mathsf{i}')$ tenemos que por un lema anterior $\ker F$ es una congruencia sobre $(L, \mathsf{s}, \mathsf{i})$ lo cual por definicion nos dice que $\ker F$ es una congruencia sobre $(L, \mathsf{s}, \mathsf{i}, 0, 1)$. ■

4.5 Reticulados complementados

Sea $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ un reticulado acotado. Dado $a \in L$, diremos que a es *complementado* cuando exista un elemento $b \in L$ (llamado *complemento de* a) tal que:

$$\begin{aligned} a \,\mathsf{s}\, b &= 1 \\ a \,\mathsf{i}\, b &= 0 \end{aligned}$$

Notese que dicho elemento b puede no ser unico, es decir a puede tener varios complementos. Recordemos que una operacion unaria sobre un conjunto L es por definicion una funcion de L en L. Muchas veces si s denota una operacion unaria, entonces escribiremos x^s en lugar de $s(x)$. Por un *reticulado complementado* entederemos una 6-upla $(L, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$ tal que $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ es un reticulado acotado y ^{c} es una operacion unaria sobre L tal que

(I10) $x \,\mathsf{s}\, x^c = 1$, para cada $x \in L$

(I11) $x \,\mathsf{i}\, x^c = 0$, para cada $x \in L$

Dado un reticulado acotado $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ puede haber mas de una operacion unaria g tal que $(L, \mathsf{s}, \mathsf{i}, g, 0, 1)$ resulte un reticulado complementado. Intente dar un ejemplo en el cual L tenga 5 elementos.

Reflexion Informatica

Notese que si tenemos un poset $(L, \leq)$ el cual es un reticulado en el cual hay un maximo 1 y un minimo 0 y ademas tenemos una funcion $g : L \to L$ tal que

$$\begin{aligned} \sup\{x, g(x)\} &= 1 \\ \inf\{x, g(x)\} &= 0 \end{aligned}$$

para cada $x \in L$, entonces podemos definir

$$\begin{array}{rcl} \mathsf{s} : L^2 & \rightarrow & L \\ (a,b) & \rightarrow & \sup(\{a,b\}) \end{array} \qquad\qquad \begin{array}{rcl} \mathsf{i} : L^2 & \rightarrow & L \\ (a,b) & \rightarrow & \inf(\{a,b\}) \end{array}$$

y se obtiene que $(L, \mathsf{s}, \mathsf{i}, g, 0, 1)$ es un reticulado complementado. Ademas en virtud del Teorema de Dedekind todo reticulado complementado se obtiene de esta forma. O sea que a nivel de informacion, tener un poset que es un reticulado con 0 y 1 junto con una operacion unaria que da complementos, es exactamente lo mismo que tener un reticulado acotado

4.5.1 Subreticulados complementados

Dados reticulados complementados $(L, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$ y $(L', \mathsf{s}', \mathsf{i}', ^{c'}, 0', 1')$ diremos que $(L, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$ *es un subreticulado complementado de* $(L', \mathsf{s}', \mathsf{i}', ^{c'}, 0', 1')$ si se dan las siguientes condiciones

(1) $L \subseteq L'$

(2) $0 = 0'$ y $1 = 1'$

(3) $\mathsf{s} = \mathsf{s}'|_{L \times L}$, $\mathsf{i} = \mathsf{i}'|_{L \times L}$ y $^c = {}^{c'}|_L$

Tal como para el caso de reticulados como ternas o de reticulados acotados tenemos el concepto de subuniverso el cual definimos a continuacion:

Sea $(L, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$ un reticulado complementado. Un conjunto $S \subseteq L$ es llamado *subuniverso* de $(L, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$ si $0, 1 \in S$ y ademas S es cerrado bajo las operaciones s, i y ^{c} (i.e. $x \mathrel{\mathsf{s}} y$, $x \mathrel{\mathsf{i}} y$, $x^c \in S$, para todo $x, y \in S$). Es importante notar que si bien los conceptos de subreticulado complementado y subuniverso estan muy relacionados, se trata de conceptos diferentes ya que los subreticulados complementados de $(L, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$ son reticulados complementados, es decir 6-uplas y los subuniversos de $(L, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$ son conjuntos, por lo cual no son 6-uplas.

Es facil de chequear que si S es un subuniverso de $(L, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$, entonces $(S, \mathsf{s}|_{S \times S}, \mathsf{i}|_{S \times S}, {}^c|_S, 0, 1)$ es un subreticulado complementado de $(L, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$ y que todo subreticulado complementado de $(L, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$ se obtiene en esta forma. Es decir, hay una biyeccion entre el conjunto de los subreticulados complementados de $(L, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$ y el conjunto de los subuniversos de $(L, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$ (cual es?). Dicho de manera mas rapida: los subuniversos de $(L, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$ son ni mas ni menos que los universos de los subreticulados complementados de $(L, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$.

4.5.2 Homomorfismos de reticulados complementados

Sean $(L, \mathsf{s}, \mathsf{i},^c, 0, 1)$ y $(L', \mathsf{s}', \mathsf{i}',^{c'}, 0', 1')$ reticulados complementados. Una funcion $F : L \to L'$ sera llamada un *homomorfismo de* $(L, \mathsf{s}, \mathsf{i},^c, 0, 1)$ *en* $(L', \mathsf{s}', \mathsf{i}',^{c'}, 0', 1')$ si para todo $x, y \in L$ se cumple que

$$\begin{array}{rcl} F(x \mathsf{ s } y) & = & F(x) \mathsf{ s}' F(y) \\ F(x \mathsf{ i } y) & = & F(x) \mathsf{ i}' F(y) \\ F(x^c) & = & F(x)^{c'} \\ F(0) & = & 0' \\ F(1) & = & 1' \end{array}$$

Un homomorfismo de $(L, \mathsf{s}, \mathsf{i},^c, 0, 1)$ en $(L', \mathsf{s}', \mathsf{i}',^{c'}, 0', 1')$ sera llamado *isomorfismo* cuando sea biyectivo y su inversa sea un homomorfismo. Como es usual usaremos el simbolo $\cong$ para denotar la relacion de isomorfismo. Escribiremos "Sea $F : (L, \mathsf{s}, \mathsf{i},^c, 0, 1) \to (L', \mathsf{s}', \mathsf{i}',^{c'}, 0', 1')$ un homomorfismo" para expresar que F es un homomorfismo de $(L, \mathsf{s}, \mathsf{i},^c, 0, 1)$ en $(L', \mathsf{s}', \mathsf{i}',^{c'}, 0', 1')$. No hay que confundirse al leer esta notacion y pensar que F es una funcion cuyo dominio es $(L, \mathsf{s}, \mathsf{i},^c, 0, 1)$, lo cual por otra parte no tiene sentido ya que el dominio de una funcion nunca puede ser una 6-upla!

Lemma 159 *Si* $F : (L, \mathsf{s}, \mathsf{i},^c, 0, 1) \to (L', \mathsf{s}', \mathsf{i}',^{c'}, 0', 1')$ *un homomorfismo biyectivo, entonces* F *es un isomorfismo*

Proof. Es dejada al lector. ■

Lemma 160 *Si* $F : (L, \mathsf{s}, \mathsf{i},^c, 0, 1) \to (L', \mathsf{s}', \mathsf{i}',^{c'}, 0', 1')$ *es un homomorfismo, entonces* I_F *es un subuniverso de* $(L', \mathsf{s}', \mathsf{i}',^{c'}, 0', 1')$. *Es decir que* F *es tambien un homomorfismo de* $(L, \mathsf{s}, \mathsf{i},^c, 0, 1)$ *en* $(I_F, \mathsf{s}'|_{I_F \times I_F}, \mathsf{i}'|_{I_F \times I_F},^{c'}, 0', 1')$

Proof. Es dejada al lector. ■

4.5.3 Congruencias de reticulados complementados

Sea $(L, \mathsf{s}, \mathsf{i},^c, 0, 1)$ un reticulado complementado. Una *congruencia sobre* $(L, \mathsf{s}, \mathsf{i},^c, 0, 1)$ sera una relacion de equivalencia sobre L la cual cumpla:

(1) θ es una congruencia sobre $(L, \mathsf{s}, \mathsf{i}, 0, 1)$

(2) $x/\theta = y/\theta$ implica $x^c/\theta = y^c/\theta$

Las condiciones anteriores nos permiten definir sobre L/θ dos operaciones binarias $\tilde{\mathsf{s}}$ e $\tilde{\imath}$, y una operacion unaria $^{\tilde{c}}$ de la siguiente manera:

$$\begin{array}{rcl} x/\theta \; \tilde{\mathsf{s}} \; y/\theta & = & (x \mathsf{ s } y)/\theta \\ x/\theta \; \tilde{\imath} \; y/\theta & = & (x \mathsf{ i } y)/\theta \\ (x/\theta)^{\tilde{c}} & = & x^c/\theta \end{array}$$

La 6-upla $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath},^{\tilde{c}}, 0/\theta, 1/\theta)$ es llamada el *cociente de* $(L, \mathsf{s}, \mathsf{i},^c, 0, 1)$ *sobre* θ y la denotaremos con $(L, \mathsf{s}, \mathsf{i},^c, 0, 1)/\theta$. Tal como era de esperar tenemos entonces

Lemma 161 *Sea* $(L, \mathsf{s}, \mathsf{i},^c, 0, 1)$ *un reticulado complementado y sea* θ *una congruencia sobre* $(L, \mathsf{s}, \mathsf{i},^c, 0, 1)$.

(a) $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath},^{\tilde{c}}, 0/\theta, 1/\theta)$ *es un reticulado complementado.*

(b) π_θ *es un homomorfismo de* $(L, \mathsf{s}, \mathsf{i},^c, 0, 1)$ *en* $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath},^{\tilde{c}}, 0/\theta, 1/\theta)$ *cuyo nucleo es* θ.

Proof. (a) Por un lema anterior ya sabemos que $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath}, 0/\theta, 1/\theta)$ es un reticulado acotado. Es decir que solo nos falta ver que $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath},^{\tilde{c}}, 0/\theta, 1/\theta)$ sarisface las identidades (I10) y (I11). Veamos por ejemplo que satisface la (I10). Sea x/θ un elemento cualquiera de L/θ. Ya que $(L, \mathsf{s}, \mathsf{i},^c, 0, 1)$ satisface la (I10), tenemos que $x \mathsf{s} x^c = 1$. O sea que $(x \mathsf{s} x^c)/\theta = 1/\theta$ y por lo tanto $x/\theta \tilde{\mathsf{s}} x^c/\theta = 1/\theta$. Pero por definicion de $^{\tilde{c}}$ tenemos que $(x/\theta)^{\tilde{c}} = x^c/\theta$, lo cual nos dice que $x/\theta \tilde{\mathsf{s}} (x/\theta)^{\tilde{c}} = 1/\theta$. Dejamos al lector ver que $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath},^{\tilde{c}}, 0/\theta, 1/\theta)$ sarisface la identidad (I11)

(b) Por el Lema 157 tenemos que π_θ es un homomorfismo de $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ en $(L/\theta, \tilde{\mathsf{s}}, \tilde{\imath}, 0/\theta, 1/\theta)$ cuyo nucleo es θ. Notese que por definicion de $^{\tilde{c}}$ tenemos que $x^c/\theta = (x/\theta)^{\tilde{c}}$, es decir $\pi_\theta(x^c) = (\pi_\theta(x))^{\tilde{c}}$, cualquiera sea $x \in L$ ■

Lemma 162 *Si* $F : (L, \mathsf{s}, \mathsf{i},^c, 0, 1) \to (L', \mathsf{s}', \mathsf{i}',^{c'}, 0', 1')$ *es un homomorfismo de reticulados complementados, entonces* $\ker F$ *es una congruencia sobre* $(L, \mathsf{s}, \mathsf{i},^c, 0, 1)$
Proof. *Ya que* F *es un homomorfismo de* $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ *en* $(L', \mathsf{s}', \mathsf{i}', 0', 1')$ *tenemos que por un lema anterior* $\ker F$ *es una congruencia sobre* $(L, \mathsf{s}, \mathsf{i}, 0, 1)$. *Es decir que solo falta probar que para todos* $x, y \in L$, *se tiene que* $x/\ker F = y/\ker F$ *implica* $x^c/\ker F = y^c/\ker F$, *lo cual es dejado al lector* ■

4.6 Algebras de Boole

Un reticulado $(L, \mathsf{s}, \mathsf{i})$ se llamara *distributivo* cuando cumpla la siguiente identidad

Dis_1 $x \mathsf{i} (y \mathsf{s} z) = (x \mathsf{i} y) \mathsf{s} (x \mathsf{i} z)$, cualesquiera sean $x, y, z \in L$

Diremos que un reticulado acotado $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ (resp. complementado $(L, \mathsf{s}, \mathsf{i},^c, 0, 1)$) es *distributivo* cuando $(L, \mathsf{s}, \mathsf{i})$ lo sea. Consideremos la distributividad dual a Dis_1, es decir

Dis_2 $x \mathsf{s} (y \mathsf{i} z) = (x \mathsf{s} y) \mathsf{i} (x \mathsf{s} z)$, cualesquiera sean $x, y, z \in L$

Lemma 163 *Sea* $(L, \mathsf{s}, \mathsf{i})$ *un reticulado. Entonces* $(L, \mathsf{s}, \mathsf{i})$ *satisface* Dis_1 *sii* $(L, \mathsf{s}, \mathsf{i})$ *satisface* Dis_2

Proof. Supongamos $(L, \mathsf{s}, \mathsf{i})$ satisface Dis_1. Sean $a, b, c \in L$ elementos fijos. Por Dis_1 tenemos que

$$(a \mathsf{\ s\ } b) \mathsf{\ i\ } (a \mathsf{\ s\ } c) = ((a \mathsf{\ s\ } b) \mathsf{\ i\ } a) \mathsf{\ s\ } ((a \mathsf{\ s\ } b) \mathsf{\ i\ } c)$$

Pero por conmutatividad tenemos que

$$((a \mathsf{\ s\ } b) \mathsf{\ i\ } a) \mathsf{\ s\ } ((a \mathsf{\ s\ } b) \mathsf{\ i\ } c) = (a \mathsf{\ i\ } (a \mathsf{\ s\ } b)) \mathsf{\ s\ } (c \mathsf{\ i\ } (a \mathsf{\ s\ } b))$$

Por (I7) tenemos que $a \mathsf{\ i\ } (a \mathsf{\ s\ } b) = a$ y por Dis_1 tenemos que $c \mathsf{\ i\ } (a \mathsf{\ s\ } b) = (c \mathsf{\ i\ } a) \mathsf{\ s\ } (c \mathsf{\ i\ } b)$ por lo cual

$$(a \mathsf{\ i\ } (a \mathsf{\ s\ } b)) \mathsf{\ s\ } (c \mathsf{\ i\ } (a \mathsf{\ s\ } b)) = a \mathsf{\ s\ } ((c \mathsf{\ i\ } a) \mathsf{\ s\ } (c \mathsf{\ i\ } b))$$

Por asociatividad tenemos que

$$a \mathsf{\ s\ } ((c \mathsf{\ i\ } a) \mathsf{\ s\ } (c \mathsf{\ i\ } b)) = (a \mathsf{\ s\ } (c \mathsf{\ i\ } a)) \mathsf{\ s\ } (c \mathsf{\ i\ } b)$$

Pero por conmutatividad tenemos que

$$(a \mathsf{\ s\ } (c \mathsf{\ i\ } a)) \mathsf{\ s\ } (c \mathsf{\ i\ } b) = (a \mathsf{\ s\ } (a \mathsf{\ i\ } c)) \mathsf{\ s\ } (b \mathsf{\ i\ } c)$$

Lo cual por (I6) nos dice que

$$(a \mathsf{\ s\ } (a \mathsf{\ i\ } c)) \mathsf{\ s\ } (b \mathsf{\ i\ } c) = a \mathsf{\ s\ } (b \mathsf{\ i\ } c)$$

Por transitividad de la igualdad, las igualdades anteriores nos dicen que

$$a \mathsf{\ s\ } (b \mathsf{\ i\ } c) = (a \mathsf{\ s\ } b) \mathsf{\ i\ } (a \mathsf{\ s\ } c)$$

Pero a, b, c eran elementos arbitrarios por lo que hemos probado que vale Dis_2.
∎

Ejercicio: Use la prueba del lema anterior para hacer un algoritmo el cual tome de entrada un reticulado acotado $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ y elementos $x, y, z \in L$ tales que $y \neq z$ son complementos de x, y de como salida elementos a, b, c tales que $a \mathsf{\ i\ } (b \mathsf{\ s\ } c) \neq (a \mathsf{\ i\ } b) \mathsf{\ s\ } (a \mathsf{\ i\ } c)$

Por un *Algebra de Boole* entenderemos un reticulado complementado que es distributivo. Algunos ejemplos:

E1 Dado un conjunto no vacio X, la 6-upla $(X, \cup, \cap, ^c, \emptyset, X)$ es un algebra de Boole

Para probar algunas propiedades fundamentales de un algebra de Boole necesitaremos el siguiente

Lemma 164 *Si* $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ *un reticulado acotado y distributivo, entonces todo elemento tiene a lo sumo un complemento. Es decir, si* $x \mathsf{s} u = x \mathsf{s} v = 1$ *y* $x \mathsf{i} u = x \mathsf{i} v = 0$*, entonces* $u = v$*, cualesquiera sean* $x, u, v \in L$*.*

Proof. Sean $a, b, c \in L$ elementos fijos. Supongamos que

$$\begin{aligned} a \mathsf{s} b &= a \mathsf{s} c = 1 \\ a \mathsf{i} b &= a \mathsf{i} c = 0 \end{aligned}$$

(es decir b y c son ambos complementos de a). Veremos que entonces $b = c$. Notese que

$$b = b \mathsf{i} 1 = b \mathsf{i} (a \mathsf{s} c) = (b \mathsf{i} a) \mathsf{s} (b \mathsf{i} c) = 0 \mathsf{s} (b \mathsf{i} c) = b \mathsf{i} c$$

por lo cual $b \leq c$. Analogamente se puede probar que $c \leq b$ por lo cual $b = c$. Ya que a, b, c eran elementos cualesquiera de L, hemos probado el lema. ■

Una propiedad muy importante que se da en las algebras de Boole es

Lemma 165 *Sea* $(B, \mathsf{s}, \mathsf{i}, ^{\mathbf{c}}, 0, 1)$ *un álgebra de Boole. Cualesquiera sean* $x, y \in B$*, se tiene que* $y = (y \mathsf{i} x) \mathsf{s} (y \mathsf{i} x^c)$*.*

Proof. Sean $a, b \in B$, fijos. Se tiene que

$$b = b \mathsf{i} 1 = b \mathsf{i} (a \mathsf{s} a^c) = (b \mathsf{i} a) \mathsf{s} (b \mathsf{i} a^c)$$

Ya que a y b eran elementos cualesquiera de B, hemos probado el lema. ■

Theorem 166 *Sea* $(L, \mathsf{s}, \mathsf{i}, ^{\mathbf{c}}, 0, 1)$ *un álgebra de Boole y sean* $a, b \in B$*. Se tiene que:*

(1) $(a \mathsf{i} b)^c = a^c \mathsf{s} b^c$

(2) $(a \mathsf{s} b)^c = a^c \mathsf{i} b^c$

(3) $a^{cc} = a$

(4) $a \mathsf{i} b = 0$ *si y solo si* $b \leq a^c$

(5) $a \leq b$ *si y solo si* $b^c \leq a^c$

Proof. (1) Es facil ver que $a^c \mathsf{s} b^c$ es un complemento de $a \mathsf{i} b$ (hacer!). Pero ya que $(L, \mathsf{s}, \mathsf{i}, ^{\mathbf{c}}, 0, 1)$ es un reticulado complementado, tenemos que $(a \mathsf{i} b)^c$ es un complemento de $a \mathsf{i} b$. El Lema 164 nos dice que $(a \mathsf{i} b)^c$ y $a^c \mathsf{s} b^c$ deben ser iguales.

(2) y (3) se prueban en forma similar (hacer!)

(4) Supongamos $a \,\mathsf{i}\, b = 0$. Se tiene

$$\begin{aligned} b &= (b \,\mathsf{i}\, a) \,\mathsf{s}\, (b \,\mathsf{i}\, a^c) \\ &= (a \,\mathsf{i}\, b) \,\mathsf{s}\, (b \,\mathsf{i}\, a^c) \\ &= 0 \,\mathsf{s}\, (b \,\mathsf{i}\, a^c) \\ &= (b \,\mathsf{i}\, a^c) \end{aligned}$$

lo cual dice que $b \leq a^c$. Supongamos $b \leq a^c$. Entonces $a \,\mathsf{i}\, b \leq a \,\mathsf{i}\, a^c = 0$ por lo cual $a \,\mathsf{i}\, b = 0$.

(5) Supongamos $a \leq b$. Entonces $a \,\mathsf{i}\, b = a$, lo cual por (1) nos dice que $a^c \,\mathsf{s}\, b^c = a^c$ obteniendo que $b^c \leq a^c$. La resiproca es dejada al lector (hint: use (3)) ■

4.7 Teoremas del filtro primo y de Rasiowa Sikorski

Un *filtro* de un reticulado $(L, \mathsf{s}, \mathsf{i})$ sera un subconjunto $F \subseteq L$ tal que:

(1) $F \neq \emptyset$

(2) $x, y \in F \Rightarrow x \,\mathsf{i}\, y \in F$

(3) $x \in F$ y $x \leq y \Rightarrow y \in F$

El nombre "filtro" es inspirado por la propiedad (3) ya que si un filtro o colador atrapa a cierto objeto x, entonces claramente atrapara a todos los objetos mas grandes que x.

Dado un conjunto $S \subseteq L$, denotemos con $[S)$ el siguiente conjunto

$$\{y \in L : y \geq s_1 \,\mathsf{i}\, ... \,\mathsf{i}\, s_n, \text{ para algunos } s_1, ..., s_n \in S, \, n \geq 1\}$$

Lemma 167 *Supongamos S es no vacio. Entonces $[S)$ es un filtro. Mas aun si F es un filtro y $F \supseteq S$, entonces $F \supseteq [S)$. Es decir, $[S)$ es el menor filtro que contiene a S.*

Proof. Ya que $S \subseteq [S)$, tenemos que $[S) \neq \emptyset$. Claramente $[S)$ cumple la propiedad (3). Veamos cumple la (2). Si $y \geq s_1 \,\mathsf{i}\, s_2 \,\mathsf{i}\, ... \,\mathsf{i}\, s_n$ y $z \geq t_1 \,\mathsf{i}\, t_2 \,\mathsf{i}\, ... \,\mathsf{i}\, t_m$, con $s_1, s_2, ..., s_n, \, t_1, t_2, ..., t_m \in S$, entonces

$$y \,\mathsf{i}\, z \geq s_1 \,\mathsf{i}\, s_2 \,\mathsf{i}\, ... \,\mathsf{i}\, s_n \,\mathsf{i}\, t_1 \,\mathsf{i}\, t_2 \,\mathsf{i}\, ... \,\mathsf{i}\, t_m$$

lo cual prueba (2). ■

Llamaremos a $[S)$ el *filtro generado por S*. Cuando S es finito, ya que existe $\inf S$, es claro que $[S) = \{y \in L : y \geq \inf S\}$. Cuando S es infinito y existe $\inf S$, en muchos casos se dara que $[S) = \{y \in L : y \geq \inf S\}$ o que $[S) = \{y \in L : y > \inf S\}$, pero no necesariamente esto sucedera siempre. Por ejemplo:

- Sea $\mathbf{L} = (\mathcal{P}(\mathbf{N}), \cup, \cap)$ y sea $S = \{\mathbf{N} - \{n\} : n \in \mathbf{N}\}$. Es facil ver que $\inf S = \emptyset$ y que $[S) = \{A \in \mathcal{P}(\mathbf{N}) : \mathbf{N} - A \text{ es finito}\}$ por lo cual no se da que $[S) = \{y \in L : y \geq \inf S\}$ o que $[S) = \{y \in L : y > \inf S\}$

En general, si $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ es un reticulado acotado, diremos que F es un *filtro* de $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ cuando F sea un filtro de $(L, \mathsf{s}, \mathsf{i})$. Lo mismo sucedera con el concepto de filtro de un reticulado complementado $(L, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$

Sea $(P, \leq)$ un poset. Un subconjunto $C \subseteq P$ sera llamado una *cadena* si para cada $x, y \in C$, se tiene que $x \leq y$ o $y \leq x$. Por ejemplo

(E1) $\{1, 10, 40, 600\}$ y $\{2^n : n \in \mathbf{N}\}$ son cadenas del poset $(\mathbf{N}, |)$

(E2) $\{-3, 5, 2\}$ y el intervalo $[2, 3]$ son cadenas del poset $(\mathbf{R}, \leq)$. De hecho todo subconjunto de $\mathbf{R}$ es una cadena de $(\mathbf{R}, \leq)$

(E3) $C = \{[0, n] : n \in \mathbf{N}\}$ es una cadena del poset $(\mathcal{P}(\mathbf{R}), \subseteq)$. Notese que cada elemento de C es un conjunto (i.e. un intervalo).

Es importante notar que las cadenas pueden ser infinitas y que dada una cadena infinita C puede no existir una infinitupla $(c_1, c_2, ...)$ tal que $C = \{c_n : n \in \mathbf{N}\}$. Este es el caso de la cadena $[0, 1]$ del poset $(\mathbf{R}, \leq)$, ya que el bien conocido argumento diagonal de Cantor nos dice que no existe una manera de enumerar los elementos del intervalo $[0, 1]$. Esto nos obliga a pensar con cierta madurez a las cadenas y no caer en la falacia de pensar que sus elementos forman necesariamente una "filita discreta".

Tambien es importante para entender la prueba del Teorema del Filtro Primo que viene a continuacion, imaginar las cadenas de posets que sus elementos son conjuntos y su orden es la inclusion, es decir dichas cadenas seran un conjunto de conjuntos C con la propiedad que dados dos cualesquiera elementos de C siempre alguno contiene al otro. Un ejemplo de este tipo de cadenas es dado en (E3). Otro ejemplo:

(E4) Sea $\mathcal{F} = \{F : F \text{ es un filtro del reticulado } (\mathbf{N}, mcm, mcd)\}$. Notar que dado $n \in \mathbf{N}$, el conjunto $\{x \in \mathbf{N} : n|x\}$ es un filtro de $(\mathbf{N}, mcm, mcd)\}$. Ya que $\mathcal{F}$ es no vacio tenemos que $(\mathcal{F}, \subseteq)$ es un poset. Entonces

$$C = \{\{x \in \mathbf{N} : n|x\} : n \text{ es potencia de } 2\}$$

es una cadena de $(\mathcal{F}, \subseteq)$.

El siguiente resultado es una herramienta fundamental en el algebra moderna.

Lemma 168 (Lema de Zorn) *Sea $(P, \leq)$ un poset y supongamos cada cadena de $(P, \leq)$ tiene al menos una cota superior. Entonces hay un elemento maximal en $(P, \leq)$.*

Obviamente en cada poset con universo finito hay al menos un elemento maximal. O sea que el Lema de Zorn es interesante para el caso en que P es un conjunto infinito. Un argumento para creer en la veracidad del lema podria ser el siguiente razonamiento por el absurdo. Supongamos que $(P, \leq)$ es un poset en el cual cada cadena tiene al menos una cota superior y supongamos ademas que no hay elementos maximales en $(P, \leq)$. Tomemos $x_0 \in P$ un elemento cualquiera. Ya que x_0 no es maximal, hay un $x_1 \in P$ tal que $x_0 < x_1$. Iterando esta idea vemos que debe haber elementos $x_2, x_3, ...$ tales que:

$$x_0 < x_1 < x_2 < x_3 < \cdots$$

Pero $\{x_0, x_1, x_2, x_3, ...\}$ es una cadena por lo cual hay al menos una cota superior de ella en $(P, \leq)$. Sea x_ω una de tales cotas.

Un filtro F de un reticulado $(L, \mathsf{s}, \mathsf{i})$ sera llamado *primo* cuando se cumplan:

(1) $F \neq L$

(2) $x \mathsf{\ s\ } y \in F \Rightarrow x \in F$ o $y \in F$.

Algunos ejemplos:

E1 Todo filtro de $(\mathbf{R}, \max, \min)$, distinto de $\mathbf{R}$, es primo (justificar)

E2 Sea $B = \{X \subseteq \omega : X$ es finito o $\omega - X$ es finito$\}$. Como vimos anteriormente B es cerrado bajo las operaciones $\cup$ y $\cap$. Sea $P = \{X \subseteq \omega : \omega - X$ es finito$\}$. Entonces P es un filtro primo de $(B, \cup, \cap)$.

Theorem 169 (Teorema del Filtro Primo) *Sea $(L, \mathsf{s}, \mathsf{i})$ un reticulado distributivo y F un filtro. Supongamos $x_0 \in L - F$. Entonces hay un filtro primo P tal que $x_0 \notin P$ y $F \subseteq P$.*

Proof. Sea

$$\mathcal{F} = \{F_1 : F_1 \text{ es un filtro, } x_0 \notin F_1 \text{ y } F \subseteq F_1\}.$$

Notese que $\mathcal{F} \neq \emptyset$, por lo cual $(\mathcal{F}, \subseteq)$ es un poset. Veamos que cada cadena en $(\mathcal{F}, \subseteq)$ tiene una cota superior. Sea C una cadena. Si $C = \emptyset$, entonces cualquier elemento de $\mathcal{F}$ es cota de C. Supongamos entonces $C \neq \emptyset$. Sea

$$G = \{x : x \in F_1, \text{ para algun } F_1 \in C\}.$$

Veamos que G es un filtro. Es claro que G es no vacio. Supongamos que $x, y \in G$. Sean $F_1, F_2 \in \mathcal{F}$ tales que $x \in F_1$ y $y \in F_2$. Si $F_1 \subseteq F_2$, entonces ya que F_2 es un filtro tenemos que $x \mathsf{i} y \in F_2 \subseteq G$. Si $F_2 \subseteq F_1$, entonces tenemos que $x \mathsf{i} y \in F_1 \subseteq G$. Ya que C es una cadena, tenemos que siempre $x \mathsf{i} y \in G$. En forma analoga se prueba la propiedad restante por lo cual tenemos que G es un filtro. Ademas $x_0 \notin G$, por lo que $G \in \mathcal{F}$ es cota superior de C. Por el lema de Zorn, $(\mathcal{F}, \subseteq)$ tiene un elemento maximal P. Veamos que P es un filtro primo. Supongamos $x \mathsf{s} y \in P$ y $x, y \notin P$. Notese que $[P \cup \{x\})$ es un filtro el cual contiene propiamente a P. Entonces ya que P es un elemento maximal de $(\mathcal{F}, \subseteq)$, tenemos que $x_0 \in [P \cup \{x\})$. Analogamente tenemos que $x_0 \in [P \cup \{y\})$. Ya que $x_0 \in [P \cup \{x\})$, tenemos que hay elementos $p_1, ..., p_n \in P$, tales que

$$x_0 \geq p_1 \mathsf{i} ... \mathsf{i} p_n \mathsf{i} x$$

(se deja como ejercicio justificar esto). Ya que $x_0 \in [P \cup \{y\})$, tenemos que hay elementos $q_1, ..., q_m \in P$, tales que

$$x_0 \geq q_1 \mathsf{i} ... \mathsf{i} q_m \mathsf{i} y$$

Si llamamos p al siguiente elemento de P

$$p_1 \mathsf{i} ... \mathsf{i} p_n \mathsf{i} q_1 \mathsf{i} ... \mathsf{i} q_m$$

tenemos que

$$\begin{aligned} x_0 &\geq p \mathsf{i} x \\ x_0 &\geq p \mathsf{i} y \end{aligned}$$

Se tiene entonces que $x_0 \geq (p \mathsf{i} x) \mathsf{s} (p \mathsf{i} y) = p \mathsf{i} (x \mathsf{s} y) \in P$, lo cual es absurdo ya que $x_0 \notin P$. ∎

Corollary 170 *Sea $(L, \mathsf{s}, \mathsf{i}, 0, 1)$ un reticulado acotado distributivo. Si $\emptyset \neq S \subseteq L$ es tal que $s_1 \mathsf{i} s_2 \mathsf{i} ... \mathsf{i} s_n \neq 0$, para cada $s_1, ..., s_n \in S$, entonces hay un filtro primo que contiene a S.*

Proof. Notese que $[S) \neq L$ por lo cual se puede aplicar el Teorema del filtro primo. ∎

Lemma 171 *Sea $(B, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$ un algebra de Boole. Entonces para un filtro $F \subsetneq B$ las siguientes son equivalentes:*

(1) F es primo

(2) $x \in F$ o $x^c \in F$, para cada $x \in B$.

Proof. (1)⇒(2). Ya que $x \mathsf{s} x^c = 1 \in F$, (2) se cumple si F es primo.

(2)⇒(1). Ya sabemos por hipotesis que F es un filtro y que $F \neq B$. Supongamos que $x \mathsf{s} y \in F$ y que $x \notin F$. Entonces por (2), $x^c \in F$ y por lo tanto tenemos que

$$y \geq x^c \mathsf{i} y = (x^c \mathsf{i} x) \mathsf{s} (x^c \mathsf{i} y) = x^c \mathsf{i} (x \mathsf{s} y) \in F,$$

lo cual dice que $y \in F$. ∎

Necesitaremos el siguiente lema.

Lemma 172 *Sea* $(B, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$ *un algebra de Boole. Supongamos que* $b \neq 0$ *y* $a = \inf A$*, con* $A \subseteq B$*. Entonces si* $b \mathsf{\ i\ } a = 0$*, existe un* $e \in A$ *tal que* $b \mathsf{\ i\ } e^c \neq 0$*.*

Proof. Supongamos que para cada $e \in A$, tengamos que $b \mathsf{\ i\ } e^c = 0$. Entonces tenemos que para cada $e \in A$,

$$b = b \mathsf{\ i\ } (e \mathsf{\ s\ } e^c) = (b \mathsf{\ i\ } e) \mathsf{\ s\ } (b \mathsf{\ i\ } e^c) = b \mathsf{\ i\ } e,$$

lo cual nos dice que b es cota inferior de A. Pero entonces $b \leq a$, por lo cual $b = b \mathsf{\ i\ } a = 0$, contradiciendo la hipotesis. ■

Es claro que si P es un filtro primo de un algebra de Boole $(B, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$, entonces cualquiera sea el conjunto finito S contenido en P, se tiene que $\inf S \in P$. Cuando tomamos un subconjunto $S \subseteq P$ el cual es infinito, la cosa cambia sustancialmente. Primero cabe destacar que puede suceder que S no tenga infimo en $(B, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$. Pero tambien puede pasar que S tenga infimo pero que $\inf S$ no pertenesca a P. Por ejemplo, si tomamos el algebra de Boole $(B, \cup, \cap, ^c, \emptyset, \omega)$, donde

$$B = \{X \subseteq \omega : X \text{ es finito o } \omega - X \text{ es finito}\}$$

podemos observar que

$$P = \{X \subseteq \omega : \omega - X \text{ es finito}\}$$

es un filtro primo y que

$$S = \{\omega - \{n\} : n \in \omega\}$$

esta contenido en P pero $\inf S = \emptyset$ no pertenece a P.

El siguiente teorema sera clave en nuestra prueba del teorema de completitud de la logica de primer orden.

Theorem 173 *(Rasiova y Sikorski) Sea* $(B, \mathsf{s}, \mathsf{i}, ^c, 0, 1)$ *un algebra de Boole. Sea* $x \in B$*,* $x \neq 0$*. Supongamos que* $(A_1, A_2, ...)$ *es una infinitupla de subconjuntos de* B *tal que existe* $\inf(A_j)$*, para cada* $j = 1, 2....$ *Entonces hay un filtro primo* P *el cual cumple:*

(a) $x \in P$

(b) $P \supseteq A_j \Rightarrow P \ni \inf(A_j)$*, para cada* $j = 1, 2, ...$

Proof. Sean $a_j = \inf(A_j)$, $j = 1, 2, ...$ Construiremos inductivamente una infinitupla $(b_0, b_1, ...)$ de elementos de B tal que:

(1) $b_0 = x$

(2) $b_0 \mathsf{\ i\ } ... \mathsf{\ i\ } b_n \neq 0$, para cada $n \geq 0$

(3) $b_j = a_j$ o $b_j^c \in A_j$, para cada $j \geq 1$.

Definamos $b_0 = x$. Supongamos ya definimos $b_0, ..., b_n$, veamos como definir b_{n+1}. Si $(b_0 \mathsf{i} ... \mathsf{i} b_n) \mathsf{i} a_{n+1} \neq 0$, entonces definamos $b_{n+1} = a_{n+1}$. Si $(b_0 \mathsf{i} ... \mathsf{i} b_n) \mathsf{i} a_{n+1} = 0$, entonces por el lema anterior, tenemos que hay un $e \in A_{n+1}$ tal que $(b_0 \mathsf{i} ... \mathsf{i} b_n) \mathsf{i} e^c \neq 0$, lo cual nos permite definir $b_{n+1} = e^c$.

Usando (2) se puede probar que el conjunto $S = \{b_0, b_1, ...\}$ satisface la hipotesis del primer corolario del Teorema del filtro primo, por lo cual hay un filtro primo P tal que $\{b_0, b_1, ...\} \subseteq P$. Es claro que $x \in P$ y es facil chequear usando (3) que P satisface la propiedad (b). ■

Cerramos la seccion con una convencion notacional que se usara mas que nada en los ejercicios y la tombola.

Convencion notacional: Notese que hemos definido distintos tipos de estructuras (i.e. posets, reticulados como ternas, etc) y en todas ellas su primera coordenada es llamada el *universo* de dicha estructura. En general usaremos letras mayusculas en bold para denotar una estructura dada y en tal caso usaremos la convension de que su correspondiente mayuscula en italica denotara el universo de dicha estructura. Por ejemplo si decimos "sea $\mathbf{L}$ un reticulado acotado", entonces ya queda implicita la informacion de que denotaremos con L al universo de $\mathbf{L}$. Ademas deberia quedar claro que en tal caso $\mathbf{L}$ es una 5-upla.

Tambien si $\mathbf{L}'$ denota una estructura, L' denotara su universo. Similarmente si $\tilde{\mathbf{L}}$ denota una estructura, $\tilde{L}$ denotara su universo, etc.

Notese que entonces, si escribimos "Sea $F : \mathbf{L} \to \mathbf{L}'$ es un homomorfismo de reticulados complementados", estaremos suponiendo que $\mathbf{L}$ y $\mathbf{L}'$ son reticulados complementados (i.e. ciertas 6-uplas) y que F es una funcion de L en L' la cual es un homomorfismo de $\mathbf{L}$ en $\mathbf{L}'$. Aqui hay que tener cuidado ya que D_F es L y no $\mathbf{L}$ lo cual seria imposible ya que $\mathbf{L}$ no es un conjunto!

Tambien notese que si $\mathbf{L}$ denota un reticulado acotado y θ es una congruencia de $\mathbf{L}$, entonces $\mathbf{L}/\theta$ denotara el cociente de $\mathbf{L}$ sobre θ, a saber cierto reticulado acotado cuyo universo es L/θ. Es decir que $Ti(\mathbf{L}/\theta) = 5-$UPLA y $Ti(L/\theta) =$ CONJUNTO

5 Estructuras y su lenguaje elemental asociado

Como ya vimos en la seccion anterior, hay distintos tipos de estructuras a las cuales se las puede estudiar usando metodos similares. Con cada tipo de estructura tenemos asociado cierto tipo de enunciados o sentencias que llamaremos *formulas elementales*. Tambien una ves que fijemos ciertas sentencias elementales que axiomaticen tal tipo de estructura, tendremos asociado un tipo de pruebas que llamaremos *elementales* ya que solo usaran dichos axiomas, deducciones muy simples y obvias de justificar con pequeñas fraces en castellano y ademas en su escritura lo concerniente a la matematica misma se escribira usando solo sentencias elementales. No daremos ahora una definicion matematica de estos conceptos pero los describiremos en forma intuitiva mediante ejemplos. Cabe destacar que los conceptos de formula elemental y prueba elemental son relativos al tipo de estructura que se esta considerando.

5.1 Formulas elementales para posets

Se construyen en forma finitaria usando simbolos de la siguiente lista:

$\forall \; \exists \; \neg \; \vee \; \wedge \; \rightarrow \leftrightarrow \; (\;) \; , \; = \; \leq$

Variables: $x, y, z, w, ...$

Nombres para elementos fijos: $a, b, c, d, ...$

Algunos ejemplos:

$(x \leq y)$

$(x = y)$

$(a \leq z)$

$(b = c)$

$\forall y(y \leq x)$

$\forall y(y \leq a)$

$\forall x(x \leq x)$

$\forall x \forall y \forall z \; ((x \leq y \wedge y \leq z) \rightarrow x \leq z)$

$\forall x \forall y((x \leq y \wedge y \leq x) \rightarrow x = y)$

$\exists x \forall y(y \leq x)$

$\neg \exists y(x \leq y \wedge \neg(y = x))$

$\forall y(y \leq x) \rightarrow \neg \exists y(x \leq y \wedge \neg(y = x))$

Notese que siempre "cuantificamos por delante", es decir que la palabra $(x \leq a)\forall x$ NO es una formula elemental de posets (aunque es claro que puede atribuirsele un sentido ya que podemos pensar que dice que a es un elemento maximo).

Para que una formula se vuelva verdadera o falsa tenemos que tener un poset concreto $(P, \leq)$ y ademas asignarles valores concretos de P a las variables libres y a los nombres de elementos fijos que figuran en dicha formula. Cuando la formula no tiene variables libres diremos que es una *sentencia elemental de posets*. Notese que en tal caso sera verdadera o falsa en un poset dado dependiendo solo de los valores que tomen los nombres para elementos fijos que ocurren en ella. Tambien cabe destacar que los cuantificadores siempre ranguean sobre P, es decir $\forall x$ se interpretara como $\forall x \in P$ y $\exists x$ se interpretara como $\exists x \in P$. La diferencia entre las variables y los nombres de elementos fijos es que si bien ambos pueden variar su valor los nombres de elementos fijos suelen denotar un valor fijo del poset durante todo un desarrollo o demostracion. Tampoco se cuantificaran los nombres de elementos fijos, es decir solo cuantificamos variables. O sea que $\forall a(a = x)$ no es una formula elemental.

Por ejemplo en el poset $(\mathbf{N}, |)$

- la formula $(x \leq y)$ es verdadera cuando le asignamos a x el valor 6 y a y el valor 36

- $\forall y(a \leq y)$ es verdadera cuando le asignamos a a el valor 1 y falsa si le asignamos el valor 5

- $\exists y(x \leq y \wedge \neg(y = x))$ es verdadera cuando le asignamos a x cualquier valor de $\mathbf{N}$

- La sentencia $\forall x \forall y(x \leq y \vee y \leq x)$ es falsa en $(\mathbf{N}, |)$

Tambien es bueno pensar que

- La formula $\forall y(a \leq y)$ dice que "a es un elemento minimo de $(P, \leq)$", la cual sera o no verdadera en $(P, \leq)$ dependiendo de que valor tenga asignado a

- La formula $\neg \exists y(x \leq y \wedge \neg(y = x))$ dice que "x es un elemento maximal de $(P, \leq)$", la cual sera o no verdadera en $(P, \leq)$ dependiendo de que valor tenga asignado x

5.2 Pruebas elementales de posets

Notese que el concepto de poset se puede axiomatizar con sentencias elementales de las recien descriptas, a saber, un *poset* es un par $(P, \leq)$ tal que P es un conjunto no vacio, $\leq$ es una relacion binaria sobre P y se cumplen:

1. $\forall x(x \leq x)$

2. $\forall x \forall y \forall z\ ((x \leq y \wedge y \leq z) \rightarrow x \leq z)$

3. $\forall x \forall y((x \leq y \wedge y \leq x) \rightarrow x = y)$

Muchas de las pruebas dadas sobre posets consisten en probar que cierta sentencia elemental se cumple en todos los posets. Por ejemplo la sentencia elemental

$$\mu = \forall x \forall y\,((\forall z\; z \leq x \wedge \forall z\; z \leq y) \rightarrow x = y)$$

claramente se cumple en todos los posets ya que ella expresa que si en un poset x e y son elementos maximos, entonces $x = y$. Mas aun, la prueba de este hecho puede escribirse usando solo formulas elementales de posets y ciertas aclaraciones minimas de castellano que sirven para justificar los distintos pasos en forma obvia. A este tipo de pruebas las llamaremos *pruebas elementales de posets*. A continuacion damos la prueba elemental de μ.

Prueba elemental de μ. Denotemos con a y b un par de elementos de A, fijos. Supongamos

$$(\forall z\; z \leq a \wedge \forall z\; z \leq b)$$

En particular $\forall z\; z \leq b$ nos dice que $a \leq b$ y $\forall z\; z \leq a$ nos dice que $b \leq a$, por lo cual tenemos que

$$a \leq b \wedge b \leq a$$

Pero el axioma

$$\forall x \forall y((x \leq y \wedge y \leq x) \rightarrow x = y)$$

nos dice que

$$(a \leq b \wedge b \leq a) \rightarrow a = b$$

obteniendo de esta forma que $a = b$. O sea que hemos probado que

$$(\forall z\; z \leq a \wedge \forall z\; z \leq b) \rightarrow a = b$$

Como a y b eran elementos cualesquiera, obtenemos que vale μ ■

5.3 Formulas elementales de reticulados terna

Se construyen en forma finitaria usando simbolos de la siguiente lista:

$\forall\ \exists\ \neg\ \vee\ \wedge\ \rightarrow \leftrightarrow\ (\ )\ ,\ =\ \mathsf{s}\ \ \mathsf{i}$

Variables: $x, y, z, w, ...$

Nombres para elementos fijos: $a, b, c, d, ...$

Algunos ejemplos:

$(x \,\mathsf{s}\, y = a)$

$(x = y)$

$((a \,\mathsf{s}\, z) \,\mathsf{i}\, x) = ((x \,\mathsf{i}\, y) \,\mathsf{s}\, x))$

$(b = c)$

$\forall x \forall y \forall z ((x \mathsf{~s~} y) \mathsf{~s~} z = x \mathsf{~s~} (y \mathsf{~s~} z))$

$\forall y (x \mathsf{~s~} y = a \rightarrow b = y)$

$\forall x \exists y (x \mathsf{~s~} y = a)$

$\forall x \forall y \forall z \ ((x = y \wedge y = z) \rightarrow x = z)$

$\neg \exists y (x \mathsf{~s~} y = y \wedge \neg (x = y))$

Notese que dado que estamos considerando ahora reticulados ternas, las *formulas elementales de reticulados ternas* no pueden usar el simbolo $\leq$ ya que no esta explicito en la presentacion de este tipo de estructura. Para que una formula se vuelva verdadera o falsa tenemos que tener un reticulado concreto $(L, \mathsf{s}, \mathsf{i})$ y ademas asignarles valores concretos de L a las variables libres y a los nombres de elementos fijos que figuran en dicha formula. Cuando la formula no tiene variables libres diremos que es una *sentencia elemental de reticulados ternas*. Notese que en tal caso sera verdadera en un reticulado terna dado, dependiendo solo de los valores que tomen los nombres de constantes fijas que ocurren en ella.

Por ejemplo en el reticulado $(\mathbf{N}, mcm, mcd)$

- la formula $(x \mathsf{~s~} y = y)$ es verdadera cuando le asignamos a x el valor 6 y a y el valor 36
- $\forall y (a = a \mathsf{~i~} y)$ es verdadera cuando le asignamos a a el valor 1 y falsa si le asignamos el valor 5
- La sentencia $\forall x \forall y (x \mathsf{~i~} y = y \mathsf{~s~} x)$ es falsa en $(\mathbf{N}, mcm, mcd)$

5.4 Pruebas elementales de reticulados terna

Igual que para el caso de los posets el concepto de reticulado terna fue definido con sentencias elementales de las recien descriptas, es decir:

1. $\forall x \forall y (x \mathsf{~s~} x = x)$
2. $\forall x \forall y (x \mathsf{~i~} x = x)$
3. $\forall x \forall y (x \mathsf{~s~} y = y \mathsf{~s~} x)$
4. $\forall x \forall y (x \mathsf{~i~} y = y \mathsf{~i~} x)$
5. $\forall x \forall y \forall z ((x \mathsf{~s~} y) \mathsf{~s~} z = x \mathsf{~s~} (y \mathsf{~s~} z))$
6. $\forall x \forall y \forall z ((x \mathsf{~i~} y) \mathsf{~i~} z = x \mathsf{~i~} (y \mathsf{~i~} z))$

7. $\forall x \forall y (x \mathsf{s} (x \mathsf{i} y) = x)$

8. $\forall x \forall y (x \mathsf{i} (x \mathsf{s} y) = x)$

Analogamente al caso de los posets, llamaremos *pruebas elementales de reticulados terna* a aquellas pruebas que partiendo de los axiomas anteriores demuestran cierta sentencia elemental usando en el camino solo sentencias elementales de reticulados ternas y ciertas aclaraciones minimas de castellano que sirven para justificar los distintos pasos en forma obvia.

Como ejemplo daremos a continuacion una prueba elemental de reticulados terna de la sentencia $\eta = \forall x \forall y (x \mathsf{s} y = y \to x \mathsf{i} y = x)$.

Prueba elemental de η. Denotemos con a y b un par de elementos de A, fijos. Supongamos

$$(a \mathsf{s} b = b)$$

El axioma

$$\forall x \forall y (x \mathsf{i} (x \mathsf{s} y) = x)$$

nos dice que

$$a \mathsf{i} (a \mathsf{s} b) = a$$

O sea que reemplazando en esta igualdad $a \mathsf{s} b$ por b obtenemos:

$$a \mathsf{i} b = a$$

Ya que habiamos supuesto que $a \mathsf{s} b = b$ hemos probado en realidad que

$$a \mathsf{s} b = b \to a \mathsf{i} b = a$$

Como a y b eran elementos cualesquiera, obtenemos que vale $\forall x \forall y (x \mathsf{s} y = y \to x \mathsf{i} y = x)$ ■

Encontrar pruebas elementales de reticulados terna tiene cierta dificultad ya que solo podemos usar los axiomas de reticulados terna y ademas no podemos escribir el simbolo $\leq$. De todas maneras podemos escribir $t \mathsf{s} s = s$ en lugar de $t \leq s$ y de esta manera simular en nuestras formulas elementales al simbolo $\leq$. Otro escollo para encontrar facilmente pruebas elementales de reticulados ternas es que de los axiomas no es obvio que las operaciones s e i sean supremo e infimo respecto del orden dado por la ecuacion $x \mathsf{s} y = y$. Esto puede ser resuelto si nos inspiramos en la prueba del Teorema de Dedeking pero de todas maneras las pruebas se complican un poco en su escritura.

A continuacion introduciremos un tipo nuevo de estructura reticulada para las cuales las pruebas elementales seran mas faciles de encontrar.

5.5 Reticulados cuaterna

Por un *reticulado cuaterna* entenderemos una 4-upla $(L, \mathsf{s}, \mathsf{i}, \leq)$ tal que L es un conjunto no vacio, s e i son operaciones binarias sobre L, $\leq$ es una relacion binaria sobre L y se cumplen los siguientes axiomas:

$$\mathrm{A}_{\leq R} = \forall x (x \leq x)$$

$$\mathrm{A}_{\leq T} = \forall x \forall y \forall z\ ((x \leq y \wedge y \leq z) \rightarrow x \leq z)$$

$$\mathrm{A}_{\leq A} = \forall x \forall y ((x \leq y \wedge y \leq x) \rightarrow x = y)$$

$$\mathrm{A}_{\mathsf{s}esC} = \forall x \forall y\ (x \leq x \mathsf{\ s\ } y \wedge y \leq x \mathsf{\ s\ } y)$$

$$\mathrm{A}_{\mathsf{s}\leq C} = \forall x \forall y \forall z\ \ ((x \leq z \wedge y \leq z) \rightarrow x \mathsf{\ s\ } y \leq z)$$

$$\mathrm{A}_{\mathsf{i}esC} = \forall x \forall y\ (x \mathsf{\ i\ } y \leq x \wedge x \mathsf{\ i\ } y \leq y)$$

$$\mathrm{A}_{\mathsf{i}\geq C} = \forall x \forall y \forall z\ \ ((z \leq x \wedge z \leq y) \rightarrow z \leq x \mathsf{\ i\ } y)$$

Obviamente los tres primeros nos garantizan que $(L, \leq)$ es un poset. Ademas notese que el axioma $\mathrm{A}_{\mathsf{s}esC}$ nos garantiza que cualesquiera sean $x, y \in L$ se tiene que $x \mathsf{\ s\ } y$ es cota superior del conjunto $\{x, y\}$. El axioma $\mathrm{A}_{\mathsf{s}\leq C}$ nos dice que cualesquiera sean los elementos $x, y \in L$, se tiene que $x \mathsf{\ s\ } s$ es menor o igual que que cualquier cota superior del conjunto $\{x, y\}$. Por supuesto esto nos garaniza que $x \mathsf{\ s\ } y = \sup\{x, y\}$. Similarmente los axiomas $\mathrm{A}_{\mathsf{i}esC}$ y $\mathrm{A}_{\mathsf{i}\geq C}$ garantizan que $x \mathsf{\ s\ } y = \inf\{x, y\}$.

O sea que, en virtud del teorema de Dedeking, tenemos que un reticulado cuaterna no es ni mas ni menos que una 4-upla $(L, \mathsf{s}, \mathsf{i}, \leq)$ tal que $(L, \mathsf{s}, \mathsf{i})$ es un reticulado terna y $\leq$ es su orden parcial asociado.

5.5.1 Formulas elementales de reticulados cuaterna

Se construyen en forma finitaria usando simbolos de la siguiente lista:

$\forall\ \exists\ \neg\ \vee\ \wedge\ \rightarrow \leftrightarrow\ (\ )\ ,\ =\ \mathsf{s}\ \ \mathsf{i}\ \leq$

Variables: $x, y, z, w, ...$

Nombres para elementos fijos: $a, b, c, d, ...$

Algunos ejemplos:

$(x \leq y)$

$(x = y)$

$(a \leq z)$

$(x \mathsf{\ s\ } y = a)$

$((a \mathsf{s} z) \mathsf{i} x) = ((x \mathsf{i} y) \mathsf{s} x))$

$\forall x \forall y \forall z((x \mathsf{s} y) \mathsf{s} z \leq x \mathsf{s} (y \mathsf{s} z))$

$\neg \exists y(x \mathsf{s} y = y \wedge \neg(x = y))$

$\forall x \forall y \forall z \ ((x \leq y \wedge y \leq z) \rightarrow x \leq z)$

$(\forall x \exists y(x \mathsf{s} y = a)) \rightarrow (a \leq a)$

5.5.2 Pruebas elementales de reticulados cuaterna

Tal como para los otros tipos de estructuras, llamaremos *pruebas elementales de reticulados cuaterna* a aquellas pruebas que partiendo de los axiomas $\mathrm{A}_{\leq R}$, $\mathrm{A}_{\leq A}$, $\mathrm{A}_{\leq T}$, $\mathrm{A}_{\mathsf{s} esC}$, $\mathrm{A}_{\mathsf{s}\leq C}$, $\mathrm{A}_{\mathsf{i} esC}$ y $\mathrm{A}_{\mathsf{i}\geq C}$ demuestran cierta sentencia elemental, usando en el camino solo sentencias elementales de reticulados cuaternas y ciertas aclaraciones minimas de castellano que sirven para justificar los distintos pasos en forma obvia.

Muchas de las pruebas dadas en la Guia 2 pueden adaptarse naturalmente para ser pruebas elementales de reticulados cuaterna. Para hacer esta adaptacion notese que el axioma $\mathrm{A}_{\leq A}$ puede ser usado en lugar de aplicar la regla Igualdad en Posets y similarmente los axiomas $\mathrm{A}_{\mathsf{s}\leq C}$ y $\mathrm{A}_{\mathsf{i}\geq C}$ se pueden usar en lugar de las reglas Superar un Supremo y Ser Menor o Igual que un Infimo. Por ejemplo veamos una prueba elemental de la sentencia $\rho = \forall x \forall y(x \mathsf{s} y = y \mathsf{s} x)$

Prueba elemental de ρ:. Sean $a, b \in L$ elementos fijos. Por el axioma $\mathrm{A}_{\mathsf{s} esC}$ (intanciado haciendo $x = b$ y $y = a$) tenemos que

$$b \leq b \mathsf{s} a \wedge a \leq b \mathsf{s} a$$

De lo cual sacamos obviamente que

$$a \leq b \mathsf{s} a \wedge b \leq b \mathsf{s} a$$

Ademas el axioma $\mathrm{A}_{\mathsf{s}\leq C}$ (instanciado haciendo $x = a$, $y = b$ y $z = b \mathsf{s} a$) nos dice que

$$((a \leq b \mathsf{s} a \wedge b \leq b \mathsf{s} a) \rightarrow a \mathsf{s} b \leq b \mathsf{s} a)$$

O sea que de las ultimas dos sentencias obtenemos trivialmente que

$$a \mathsf{s} b \leq b \mathsf{s} a$$

En forma analoga se puede probar que

$$b \mathsf{s} a \leq a \mathsf{s} b$$

Lo cual nos dice trivialmente que

$$a \mathsf{s} b \leq b \mathsf{s} a \wedge b \mathsf{s} a \leq a \mathsf{s} b$$

Pero el axioma $\mathrm{A}_{\leq A}$ nos dice que

$$(a \mathsf{s} b \leq b \mathsf{s} a \wedge b \mathsf{s} a \leq a \mathsf{s} b) \rightarrow a \mathsf{s} b = b \mathsf{s} a$$

De lo cual obviamente obtenemos que

$$a \mathsf{s} b = b \mathsf{s} a$$

Ya que a, b eran elementos fijos pero arbitrarios, hemos probado que

$$\forall x \forall y (x \mathsf{s} y = y \mathsf{s} x)$$

■

Por supuesto, en la parte de la prueba en la que decimos "En forma analoga se puede probar que $b \mathsf{s} a \leq a \mathsf{s} b$" deberiamos poner las lineas que corresponden para obtener la prueba elemental completa.

Ahora daremos una prueba elemental de la sentencia $\mu = \forall x \forall y (x \leq y \leftrightarrow x \mathsf{s} y = y)$.

Prueba elemental de μ:. Sean $a, b \in L$ elementos fijos. Supongamos que $a \leq b$. Probaremos que $a \mathsf{s} b = b$. Notese que por el axioma $\mathrm{A}_{\mathsf{s} esC}$ tenemos que

$$b \leq a \mathsf{s} b$$

Por el axioma $\mathrm{A}_{\mathsf{s} \leq C}$ tenemos que

$$((a \leq b \wedge b \leq b) \rightarrow a \mathsf{s} b \leq b)$$

Pero por el axioma $\mathrm{A}_{\leq R}$ tenemos que $b \leq b$ y por hipotesis tenemos que $a \leq b$ por lo cual

$$a \leq b \wedge b \leq b$$

Obviamente esto nos dice que $a \mathsf{s} b \leq b$. O sea que hemos probado

$$a \mathsf{s} b \leq b \wedge b \leq a \mathsf{s} b$$

Lo cual por el axioma $\mathrm{A}_{\leq A}$ nos dice que $a \mathsf{s} b = b$. Ya que habiamos asumido que $a \leq b$ en realidad hemos probado que

$$a \leq b \rightarrow a \mathsf{s} b = b$$

Supongamos ahora que $a \mathsf{s} b = b$. Por el axioma $\mathrm{A}_{\mathsf{s} esC}$ tenemos que $a \leq a \mathsf{s} b$. Ya que $a \mathsf{s} b = b$ obtenemos que $a \leq b$. O sea que realmente hemos probado que

$$a \mathsf{s} b = b \rightarrow a \leq b$$

Lo cual por la otra implicacion probada nos dice que

$$a \leq b \leftrightarrow a \mathsf{s} b = b$$

Ya que a, b eran elementos fijos pero arbitrarios, hemos probado que

$$\forall x \forall y (x \leq y \leftrightarrow x \mathsf{s} y = y)$$

■

5.6 Formulas elementales de reticulados complementados

Ya que los reticulados complementados son 6-uplas $(L, \mathsf{s}, \mathsf{i}, c, 0, 1)$ las formulas elementales se construyen en forma finitaria usando simbolos de la siguiente lista:

$\forall\ \exists\ \neg\ \vee\ \wedge\ \rightarrow\ \leftrightarrow\ (\ )\ ,\ =\ \mathsf{s}\ \ \mathsf{i}\ \ c\ \ 0\ \ 1$

Variables: $x, y, z, w, ...$

Nombres para elementos fijos: $a, b, c, d, ...$

Algunos ejemplos:

$(0 \mathrel{\mathsf{s}} c(y) = a)$

$(x \mathrel{\mathsf{s}} 1 = a)$

$((c(a) \mathrel{\mathsf{s}} z) \mathrel{\mathsf{i}} x) = ((x \mathrel{\mathsf{i}} y) \mathrel{\mathsf{s}} x))$

$(c(c(c(b))) = x)$

$\forall x \forall y \forall z((x \mathrel{\mathsf{s}} y) \mathrel{\mathsf{s}} z = x \mathrel{\mathsf{s}} (y \mathrel{\mathsf{s}} z))$

$(x \mathrel{\mathsf{s}} y = 1 \wedge x \mathrel{\mathsf{i}} y = 0) \rightarrow x = c(y)$

$\forall x \forall y \forall z\ ((x = y \wedge y = z) \rightarrow x = z)$

$\exists x \exists z \exists y\ c((a \mathrel{\mathsf{s}} z) \mathrel{\mathsf{i}} x)) = ((x \mathrel{\mathsf{i}} c(y)) \mathrel{\mathsf{s}} x))$

5.7 Pruebas elementales de reticulados complementados

Igual que para el caso de los reticulados terna el concepto de reticulado complementado fue definido con sentencias elementales de las recien descriptas, es decir:

1. $\forall x \forall y(x \mathrel{\mathsf{s}} x = x)$
2. $\forall x \forall y(x \mathrel{\mathsf{i}} x = x)$
3. $\forall x \forall y(x \mathrel{\mathsf{s}} y = y \mathrel{\mathsf{s}} x)$
4. $\forall x \forall y(x \mathrel{\mathsf{i}} y = y \mathrel{\mathsf{i}} x)$
5. $\forall x \forall y \forall z((x \mathrel{\mathsf{s}} y) \mathrel{\mathsf{s}} z = x \mathrel{\mathsf{s}} (y \mathrel{\mathsf{s}} z))$
6. $\forall x \forall y \forall z((x \mathrel{\mathsf{i}} y) \mathrel{\mathsf{i}} z = x \mathrel{\mathsf{i}} (y \mathrel{\mathsf{i}} z))$
7. $\forall x \forall y(x \mathrel{\mathsf{s}} (x \mathrel{\mathsf{i}} y) = x)$
8. $\forall x \forall y(x \mathrel{\mathsf{i}} (x \mathrel{\mathsf{s}} y) = x)$

9. $\forall x(0 \mathsf{s} x = x)$
10. $\forall x(x \mathsf{s} 1 = 1)$
11. $\forall x(x \mathsf{s} c(x) = 1)$
12. $\forall x(x \mathsf{i} c(x) = 0)$

Analogamente al caso de los posets, llamaremos *pruebas elementales de reticulados complementados* a aquellas pruebas que partiendo de los axiomas anteriores demuestran cierta sentencia elemental usando en el camino solo sentencias elementales de reticulados complementados y ciertas aclaraciones minimas de castellano que sirven para justificar los distintos pasos en forma obvia. Aqui tambien como en el caso de reticulados terna tenemos el problema de no poder escribir el simbolo $\leq$ en nuestras pruebas elementales de reticulados complementados y tambien el escollo de que los axiomas no hacen referencia obvia a que las operaciones s e i sean operciones supremo e infimo respecto del orden asociado. Sin envargo muchas pruebas se pueden hacer en forma natural. Por ejemplo, notar que toda prueba elemental de reticulados terna es tambien una prueba elemental de reticulados complementados, por lo cual tenemos una prueba elemental de reticulados complementados de la sentencia

$$\forall x \forall y(x \mathsf{s} y = y \rightarrow x \mathsf{i} y = x)$$

Otro ejemplo:

Prueba elemental de $\forall x(x \mathsf{i} 1 = x)$:. Sea $a \in L$ fijo. El axioma

$$\forall x(x \mathsf{s} 1 = 1)$$

nos dice que

$$a \mathsf{s} 1 = 1$$

Ya que

$$\forall x \forall y(x \mathsf{s} y = y \rightarrow x \mathsf{i} y = x)$$

(teorema ya probado) tenemos que

$$a \mathsf{s} 1 = 1 \rightarrow a \mathsf{i} 1 = a$$

O sea que

$$a \mathsf{i} 1 = a$$

Ya que a era arbitrario, hemos probado que

$$\forall x(x \mathsf{i} 1 = x)$$

■

Notese que si realmente queremos tener la prueba elemental completa de $\forall x(x \mathsf{i} 1 = x)$ deberiamos agregar en la prueba anterior las lineas correspondientes a la prueba de $\forall x \forall y(x \mathsf{s} y = y \rightarrow x \mathsf{i} y = x)$.

Dejamos al lector decribir tal como se viene haciendo desde el comienso de esta seccion los conceptos de formula elemental de reticulados acotados y prueba elemental de reticulados acotados.

5.8 Grafos

Un *grafo* es un par (G, r) donde G es un conjunto no vacio, r es una relacion binaria sobre G y se cumple que:

- $\forall x \forall y (r(x, y) \rightarrow r(y, x))$

(notese que escribimos $r(x, y)$ para expresar que $(x, y) \in r$). Hay varias presentaciones del concepto de grafo no dirigido pero el lector no tardara en darse cuenta que estas estructuras son equivalentes a las que el haya estudiado bajo el nombre de grafos no dirigidos.

5.8.1 Formulas elementales de grafos

Las formulas elementales de grafos son aquellas que se pueden construir en forma finitaria usando simbolos de la siguiente lista:

$\forall\ \exists\ \neg\ \vee\ \wedge\ \rightarrow \leftrightarrow\ (\ )\ ,\ = r$

Variables: $x, y, z, w, ...$

Nombres para elementos fijos: $a, b, c, d, ...$

Algunos ejemplos:

$r(x, y)$

$r(a, z)$

$\forall y r(a, y)$

$\forall x \forall y \forall z\ ((r(x, y) \wedge r(y, z)) \rightarrow r(x, z))$

$\exists x \forall z \neg r(x, z)$

$\forall x \exists y (r(x, y) \wedge \forall z\ r(y, z))$

$\forall x \forall y \forall z\ ((x = y \wedge y = z) \rightarrow x = z)$

5.8.2 Pruebas elementales de grafos

Llamaremos *pruebas elementales de grafos* a aquellas pruebas que partiendo del axioma $\forall x \forall y (r(x,y) \rightarrow r(y,x))$ demuestran cierta sentencia elemental de grafos usando en el camino solo sentencias elementales de grafos y ciertas aclaraciones minimas de castellano que sirven para justificar los distintos pasos en forma obvia. Es dificil encontrar pruebas elementales de grafos que no sean complicadas. Aceptando cierto grado de complejidad hay muchas. Un dato interesante es que el conjunto de sentencias elementales de grafos que tienen una prueba elemental es no recursivo, es decir no hay un procedimiento efectivo que decida si una sentencia elemental de grafos dada tiene una prueba elemental. Esto habla acerca de cuan complicada puede ser la estructura o fisonomia de las sentencias elementales de grafos que pueden ser probadas elementalmente.

5.9 Median algebras

Una *median algebra* es un par (A, M) donde A es un conjunto no vacio, M es una operacion 3-aria sobre A (i.e. $M : A^3 \rightarrow A$) y se cumplen:

1. $\forall x \forall y \forall z (M(x,y,z) = M(x,z,y))$
2. $\forall x \forall y \forall z (M(x,y,z) = M(y,z,x))$
3. $\forall x \forall y (M(x,x,y) = x)$
4. $\forall x \forall y \forall z \forall u \forall v (M(M(x,y,z),u,v)) = M(x, M(y,u,v), M(z,u,v)))$

Por ejemplo si tomamos un reticulado $(L, \mathsf{s}, \mathsf{i})$ y definimos $M(x,y,z) = (x \mathrel{\mathsf{i}} y) \mathrel{\mathsf{s}} (x \mathrel{\mathsf{i}} z) \mathrel{\mathsf{s}} (y \mathrel{\mathsf{i}} z)$, para cada $x, y, z \in L$, entonces (L, M) es una median algebra.

5.9.1 Formulas elementales de median algebras

Las formulas elementales de median algebras son aquellas que se pueden construir en forma finitaria usando simbolos de la siguiente lista:

$\forall \ \exists \ \neg \ \vee \ \wedge \ \rightarrow \leftrightarrow \ (\ ) \ , \ = M$

Variables: $x, y, z, w, ...$

Nombres para elementos fijos: $a, b, c, d, ...$

Algunos ejemplos:

$M(x,y,z) = a$

$M(a, b, M(M(M(x,y,z),u,v),x,a) = M(y,u,a)$

$\forall y(M(x,y,z)=z)$

$\forall x\exists y(M(x,y,y)=y \to \forall z\ M(a,y,z)=M(x,x,x))$

$\exists x\forall z\neg(M(a,a,a)=a \vee \exists z(M(a,y,z)=b))$

5.9.2 Pruebas elementales de median algebras

Llamaremos *pruebas elementales de median algebras* a aquellas pruebas que partiendo de los axiomas de median algebras demuestran cierta sentencia elemental de median algebras usando en el camino solo sentencias elementales de median algebras y ciertas aclaraciones minimas de castellano que sirven para justificar los distintos pasos en forma obvia. Es dificil encontrar pruebas elementales de median algebras que no sean complicadas. Un ejemplo:

Prueba elemental de $\forall x\forall y(M(x,y,y)=y)$:. Sean $a,b\in A$ fijos. Por el axioma $\forall x\forall y\forall z(M(x,y,z)=M(y,z,x))$ tenemos que

$$M(a,b,b)=M(b,b,a)$$

Por el axioma $\forall x\forall y(M(x,x,y)=x)$ tenemos que

$$M(b,b,a)=b$$

O sea que

$$M(a,b,b)=b$$

Ya que a,b eran cualesquiera, hemos probado que $\forall x\forall y(M(x,y,y)=y)$ ■

5.10 Grafos bicoloreados

Recordemos que dado un grafo (G,r), un *coloreo de* (G,r) es una asignacion de colores a cada elemento de G de manera que nunca dos elementos de G que esten reacionados tengan el mismo color. En el caso que solo usemos dos colores, le llamaremos un *bicoloreo de* (G,r). Notese que un bicoloreo puede ser representado con un subconjunto de G. Por ejemplo si el bicoloreo coloreaba a los elementos de G con dos colores, verde y rojo, podemos tomar $R=\{g\in G : g$ es rojo$\}$ y esto determina nuestro bicoloreo ya que $G-R$ sera justamente el conjunto de elemenos verdes. O sea que matematicamente hablando podemos dar la siguiente definicion. Un *bicoloreo de* (G,r) es un subconjunto R de G el cual cumple que cualesquiera sean $x,y\in G$ se tiene que

$$\text{si } (x,y)\in r,\text{ entonces } x\in R \text{ si y solo si } y\notin R$$

Esto nos inspira para dar la definicion de un nuevo tipo de estructura.

Un *grafo bicoloreado* es una terna (G,r,R), donde (G,r) es un grafo y R es un bicoloreo de (G,r). Algunos ejemplos:

- $(\{1,2\},\{(1,2),(2,1)\},\{1\})$ es un grafo bicoloreado

- Tomemos

$$\begin{array}{rcl} G & = & \omega \\ r & = & \{(x,x+1):x\in\omega\}\cup\{(x+1,x):x\in\omega\} \\ R & = & \{x\in\omega : x \text{ es par}\} \end{array}$$

  Entonces (G,r,R) es un grafo bicoloreado

- $(\{1,2,3,4\},\emptyset,R)$ es un grafo bicoloreado, cualesquera sea $R\subseteq\{1,2,3,4\}$

5.10.1 Formulas elementales de grafos bicoloreados

Para escribir las formulas elementales de grafos bicoloreados, pensaremos a R como una "relacion unaria" y escribiremos $R(x)$ para expresar que $x\in R$. Asi como cuando escribimos $r(x,y)$ pensamos "x e y estan conectados", cuando escribamos $R(x)$ pensaremos "x es rojo". Esto hace mas lejibles nuestras expresiones.

Las formulas elementales de grafos bicoloreados son aquellas que se pueden construir en forma finitaria usando simbolos de la siguiente lista:

$\forall\ \exists\ \neg\ \vee\ \wedge\ \rightarrow\ \leftrightarrow\ (\ )\ ,\ =\ r\ \ R$

Variables: $x,y,z,w,...$

Nombres para elementos fijos: $a,b,c,d,...$

Algunos ejemplos:

$R(a)\wedge r(x,y)$

$\exists z(r(a,z)\rightarrow R(z))$

$\forall y r(a,y)$

$\forall x\forall y((R(x)\wedge R(y))\rightarrow x=y)$

$\exists x\forall z(\neg R(z)\rightarrow r(x,z))$

$\forall x\forall y(r(x,y)\rightarrow(R(x)\leftrightarrow\neg R(y)))$

$\forall x\forall y\forall z\ ((x=y\wedge y=z)\rightarrow x=z)$

5.10.2 Pruebas elementales de grafos bicoloreados

Notese que los grafos bicoloreados pueden axiomatizarse con las sentencias elementales:

1. $\forall x \forall y (r(x,y) \rightarrow r(y,x))$
2. $\forall x \forall y (r(x,y) \rightarrow (R(x) \leftrightarrow \neg R(y)))$

Llamaremos *pruebas elementales de grafos bicoloreados* a aquellas pruebas que partiendo de los dos axiomas anteriores demuestran cierta sentencia elemental de grafos bicoloreados usando en el camino solo sentencias elementales de grafos bicoloreados y ciertas aclaraciones minimas de castellano que sirven para justificar los distintos pasos en forma obvia. Es dificil encontrar pruebas elementales de grafos bicoloreados que no sean complicadas. Un par de ejemplos simples:

Prueba elemental de $\forall x \neg r(x,x)$:. Sea $a \in G$ fijo. Supongamos $r(a,a)$. Por el axioma $\forall x \forall y (r(x,y) \rightarrow (R(x) \leftrightarrow \neg R(y)))$ tenemos que $(r(a,a) \rightarrow (R(a) \leftrightarrow \neg R(a))$. O sea que tenemos que

$$R(a) \leftrightarrow \neg R(a)$$

lo cual claramente es un absurdo. El absurdo proviene de suponer $r(a,a)$ lo cual nos dice que $\neg r(a,a)$. Ya que a era arbitrario, hemos probado que $\forall x \neg r(x,x)$. ■

Prueba elemental de $\forall x \forall y \forall z \neg(r(x,y) \wedge r(x,z) \wedge r(y,z))$:. Sean $a,b,c \in G$ fijos. Supongamos $r(a,b) \wedge r(a,c) \wedge r(b,c)$. Supongamos que $R(a)$. Ya que $r(a,b)$ el axioma $\forall x \forall y (r(x,y) \rightarrow (R(x) \leftrightarrow \neg R(y)))$ nos dice que se da $R(a) \leftrightarrow \neg R(b)$, de lo cual sacamos que $\neg R(b)$. Ya que $r(a,c)$ el mismo axioma nos dice que se da $R(a) \leftrightarrow \neg R(c)$, de lo cual sacamos que $\neg R(c)$. Ya que $r(b,c)$ el mismo axioma nos dice que $R(b) \leftrightarrow \neg R(c)$, de lo cual sacamos $R(b)$ (ya que se daba $\neg R(c)$). O sea que probamos que se da $R(b) \wedge \neg R(b)$, lo cual es absurdo. El absurdo proviene de suponer que se daba $R(a)$, por lo cual hemos probado que se da $\neg R(a)$. Ya que $r(a,b)$ el axioma $\forall x \forall y (r(x,y) \rightarrow (R(x) \leftrightarrow \neg R(y)))$ nos dice que se da $R(a) \leftrightarrow \neg R(b)$, de lo cual sacamos que $R(b)$ (ya que se da $\neg R(a)$). Ya que $r(a,c)$ el mismo axioma nos dice que se da $R(a) \leftrightarrow \neg R(c)$, de lo cual sacamos que $R(c)$. O sea que se da $R(b) \wedge R(c)$. Ya que $r(b,c)$ el mismo axioma nos dice que $R(b) \leftrightarrow \neg R(c)$, lo cual contradice $R(b) \wedge R(c)$. El absurdo proviene de suponer que se daba $r(a,b) \wedge r(a,c) \wedge r(b,c)$ por lo cual hemos probado que $\neg(r(a,b) \wedge r(a,c) \wedge r(b,c))$. Ya que a,b,c eran arbitrarios, hemos probado que $\forall x \forall y \forall z \neg(r(x,y) \wedge r(x,z) \wedge r(y,z))$. ■

6 Logica matematica

En la Seccion 4 nos focalizamos en aprender algebra con la intencion de volvernos lo mas "algebristas profecionales" que podamos. Para esto fuimos exigentes a la hora de delimitar y manejar nuestro lenguaje matematico y tambien a la hora de hacer pruebas pusimos mucha atencion en hacerlas "perfectas" en el sentido de que sean similares a las que haria un algebrista formado.

Pero para que hicimos esto? Muy simple: la logica matematica es *matematica aplicada* al estudio de los matematicos, su lenguaje y sus metodos de demostracion, y que mas comodo para hacer logica matematica que contar con un matematico dentro de uno mismo para estudiarlo! Tal como

- un biologo estudia la estructura y funcionamiento de los seres vivos
- un astronomo estudia los cuerpos celestes
- un fisico estudia la materia y su comportamiento

un logico matematico estudia con herramientas matematicas a los mismos matematicos en cuanto a sus caracteristicas en su roll haciendo matematica. Es decir nos interesa dar un modelo matematico que describa en forma matematica precisa el funcionamiento de un matematico en cuanto a su lenguaje y sus metodos de demostracion. Pero algo debe quedar muy claro: haremos matematica aplicada, es decir, no es nuestra intencion decirle a un matematico como debe razonar! Todo lo contrario, sabemos que los matematicos profecionales actuales razonan correctamente y que su estilo de prueba es correcto, dado el avansado estado actual de la disciplina y de sus profecionales. Simplemente los estudiaremos con herramientas matematicas para tratar de dar una descripcion matematica de su lenguaje y de sus metodos de demostracion.

Por supuesto hacer logica matematica puede ser muy dificil o escurridizo ya que como todos sabemos los matematicos tienen metodos dificiles de entender y un lenguaje verdaderamente complicado.

La forma en la que encararemos el problema sera la siguiente. En lugar de estudiar a un matematico en su actividad real crearemos un "contexto matematico simplificado" en el cual tambien tenga sentido hacer matematica profecional y luego estudiaremos a un matematico haciendo matematica en este contexto. Por supueto esto baja mucho el nivel de nuestra ambicion cientifica como logicos matematicos ya que en lugar de estudiar a los matematicos en su vida real, los estudiaremos en un contexto simplificado. Sin envargo como veremos mas adelante nuestra simplificacion no nos hara perder generalidad y los resultados obtenidos daran un modelo matematico del quehacer matematico real. Este hecho es uno de los logros mas importantes de la ciencia moderna.

Para crear este "contexto matematico simplificado" nos serviran los conceptos de lenguaje elemental y prueba elemental. Mas concretamente fijaremos un tipo de estructura, por ejemplo los reticulados cuaterna, y estudiaremos a un matematico profecional haciendo matematica en este contexto elemental. Es decir le pediremos que realice pruebas de propiedades que valgan en todos los reticulados cuaterna pero lo restringirenos en su lenguaje, es decir le pediremos que se restrinja a usar solo formulas elementales de reticulados cuaterna y que las pruebas que realice sean tambien elementales de reticulados cuaterna. El matematico rapidamente entendera la consigna y posiblemente refunfuñe un poco porque claramente lo estamos restrinjiendo mucho en relacion a su manera de hacer matematica (por ejemplo no podra hablar de filtros primos, etc). De todas maneras las posibilidades de hacer matematica profunda e interesante aun con esta restriccion son inmensas, es decir hay verdades de reticulados cuaterna que son elementales en enunciado y prueba pero son extremadamente dificiles, ingeniosas y profundas.

En este proyecto de hacer logica matematica estudiando a un matematico haciendo matematica elemental de reticulados cuaterna hay varias cosas para hacer y las establecemos a continuacion.

Programa de logica sobre reticulados cuaterna

- Dar una definicion matematica del concepto de formula elemental de reticulados cuaterna

- Dar una definicion matematica de cuando una formula elemental es verdadera en un reticulado cuaterna dado para una asignacion dada de valores de dicho reticulado a las variables libres y a los nombres de constantes fijas de la formula

- (Plato gordo) Dar un modelo matematico del concepto de prueba elemental de reticulados cuaterna. A estos objetos matematicos que modelizaran a las pruebas elementales de los matematicos los llamaremos pruebas formales de reticulados cuaterna.

- (Sublime) Intentar probar matematicamente que nuestro concepto de prueba es una correcta modelizacion matematica de la idea intuitiva de prueba elemental que todo matematico profecional posee.

Como veremos, los cuatro puntos anteriores pueden ser hechos satisfactoriamente y constituyen el comienzo de la logica matematica con cuantificadores. Cabe aclarar que la realizacion del cuarto punto es realmente sorprendente ya que es un caso de una prueba matematica rigurosa de un hecho que involucra un concepto intuitivo como lo es el de prueba elemental

Ya que la realizacion de los 4 puntos anteriores no depende en absoluto de que hayamos elejido el tipo de estructura de los reticulados cuaterna (es decir, el desarrollo que resuelve los 4 puntos anteriores para los reticulados cuaterna puede adaptarse facilmente para cualquiera de los otros tipos de estructuras descriptos en la Seccion 5), haremos las cosas con mas generalidad.

Primero, basados en la Seccion 5, generalizaremos el concepto de estructura. La generalizacion que daremos del concepto de estructura es realmente muy amplia y nos llevara mucho trabajo de entrenamiento poder manejarla con madurez y naturalidad. Luego, estableceremos para un tipo generico de estructura el programa de logica arriba escrito para el caso particular de los reticulados cuaterna. En las subsiguientes secciones nos dedicaremos a resolver dicho programa general.

6.1 Tipos

Para generalizar el concepto de estructura es conveniente primero dar definiciones generales de los conceptos de operacion y de relacion sobre un conjunto.

Sea A un conjunto y sea $n \in \mathbf{N}$. Por una *operacion n-aria sobre A* entenderemos una funcion cuyo dominio es A^n y cuya imagen esta contenida en A. Por una *relacion n-aria sobre A* entenderemos un subconjunto de A^n. Notar que por la definicion anterior una relacion unaria sobre A no es ni mas ni menos que un subconjunto de A.

Como venimos viendo hay una variedad de tipos de estructuras las cuales tienen un sentido o interes matematico claro y todas son de un formato similar, a saber uplas formadas por una primera coordenada que es un conjunto no vacio (llamado el universo de la estructura) y luego ciertas relaciones, operaciones y elementos distinguidos, dependiendo del caso. Otra cosa a notar es que para cada tipo de estructura hay ciertos simbolos fijos que usamos en forma generica para denotar sus relaciones, operaciones y elementos distinguidos. Por ejemplo:

- Para los posets usamos el simbolo $\leq$ para denotar la relacion de orden parcial en un sentido generico.

- Para el caso de los reticulados terna usamos en forma generica los simbolos s e i para denotar sus operaciones supremo e infimo

- Para el caso de los reticulados acotados usamos en forma generica los simbolos s e i para denotar sus operaciones supremo e infimo y los numerales 0 y 1 para denotar sus elementos distinguidos, a saber maximo y minimo respectivamente.

- Para el caso de los reticulados complementados usamos en forma generica los simbolos s e i para denotar sus operaciones supremo e infimo, el simbolo c para denotar su operacion 1-aria de complementacion y los numerales 0 y 1 para denotar sus elementos distinguidos, a saber maximo y minimo respectivamente.

- Para el caso de los reticulados cuaterna usamos en forma generica los simbolos s e i para denotar sus operaciones supremo e infimo y el simbolo $\leq$ para denotar su orden parcial

- Para las median algebras usamos genericamente el simbolo M para denotar su operacion 3-aria.

- Para los grafos usamos el simbolo r para denotar en forma generica la relacion binaria del grafo en cuestion.

- Para los grafos bicoloreados usamos el simbolo r para denotar en forma generica la relacion binaria del grafo y el simbolo R para denotar la relacion unaria que determina el bicoloreo

O sea que para cada tipo de estructuras se distinguen tres conjuntos de simbolos:

- un conjunto $\mathcal{R}$ formado por los simbolos que denotaran genericamente las relaciones de las estructuras

- un conjunto $\mathcal{F}$ formado por los simbolos que denotaran genericamente las operaciones de las estructuras

- un conjunto $\mathcal{C}$ formado por los simbolos que denotaran genericamente los elementos distinguidos de las estructuras

Ademas otra informacion importante que se tiene para cada tipo de estructura es la aridad de las operaciones que denotan los simbolos de $\mathcal{F}$ y la aridad de las relaciones que denotan los simbolos de $\mathcal{R}$. A esto lo representaremos con una funcion $a : \mathcal{F} \cup \mathcal{R} \to \mathbf{N}$ la cual le asocia a cada simbolo de $\mathcal{F} \cup \mathcal{R}$ la aridad del objeto que denota.

Ejemplos:

- Posets: $\mathcal{C} = \emptyset \quad \mathcal{F} = \emptyset \quad \mathcal{R} = \{\leq\} \quad a = \{(\leq, 2)\}$

- Reticulados terna: $\mathcal{C} = \emptyset \quad \mathcal{F} = \{\mathsf{s}, \mathsf{i}\} \quad \mathcal{R} = \emptyset \quad a = \{(\mathsf{s}, 2), (\mathsf{i}, 2)\}$

- Reticulados acotados: $\mathcal{C} = \{0, 1\} \quad \mathcal{F} = \{\mathsf{s}, \mathsf{i}\} \quad \mathcal{R} = \emptyset \quad a = \{(\mathsf{s}, 2), (\mathsf{i}, 2)\}$

- Reticulados comp.: $\mathcal{C} = \{0, 1\} \quad \mathcal{F} = \{\mathsf{s}, \mathsf{i}, c\} \quad \mathcal{R} = \emptyset \quad a = \{(\mathsf{s}, 2), (\mathsf{i}, 2), (c, 1)\}$

- Reticulados cuaterna: $\mathcal{C} = \emptyset \quad \mathcal{F} = \{\mathsf{s}, \mathsf{i}\} \quad \mathcal{R} = \{\leq\} \quad a = \{(\mathsf{s}, 2), (\mathsf{i}, 2), (\leq, 2)\}$

- Median algebras: $\mathcal{C} = \emptyset \quad \mathcal{F} = \{M\} \quad \mathcal{R} = \emptyset \quad a = \{(M, 3)\}$

- Grafos: $\mathcal{C} = \emptyset \quad \mathcal{F} = \emptyset \quad \mathcal{R} = \{r\} \quad a = \{(r, 2)\}$

- Grafos bicoloreados: $\mathcal{C} = \emptyset \quad \mathcal{F} = \emptyset \quad \mathcal{R} = \{r, R\} \quad a = \{(r, 2), (R, 1)\}$

Por supuesto aqui es muy importante no confundir los simbolos con las operaciones que eventualmente ellos denotan. O sea en todos los ejemplos anteriores los elementos de $\mathcal{C}$, $\mathcal{F}$ y $\mathcal{R}$ son simbolos, es decir su Ti es PALABRA. Es decir, segun el contexto si escribimos $\leq$, puede ser que nos estemos refiriendo a un orden parcial o simplemente al mismo simbolo $\leq$

Lo anterior motiva la siguiente definicion de tipo (de estructura). Antes de darla recordemos que si α, β son palabras cualesquiera, decimos que α *es subpalabra (propia) de* β cuando ($\alpha \notin \{\varepsilon, \beta\}$ y) existen palabras δ, γ tales que $\beta = \delta\alpha\gamma$.

Ahora si, nuestra definicion de tipo:

Por un *tipo* (*de primer orden*) entenderemos una 4-upla $\tau = (\mathcal{C}, \mathcal{F}, \mathcal{R}, a)$ tal que:

(1) Hay alfabetos finitos Σ_1, Σ_2 y Σ_3 tales:

  (a) $\mathcal{C} \subseteq \Sigma_1^+$, $\mathcal{F} \subseteq \Sigma_2^+$ y $\mathcal{R} \subseteq \Sigma_3^+$

  (b) Σ_1, Σ_2 y Σ_3 son disjuntos de a pares.

  (c) $\Sigma_1 \cup \Sigma_2 \cup \Sigma_3$ no contiene ningun simbolo de la lista $\forall\ \exists\ \neg\ \vee\ \wedge\ \rightarrow\ \leftrightarrow\ (\ )\ ,\ \equiv \mathsf{X}$ *0 1 ... 9* **0 1 ... 9**

(2) $a : \mathcal{F} \cup \mathcal{R} \rightarrow \mathbf{N}$ es una funcion que a cada $p \in \mathcal{F} \cup \mathcal{R}$ le asocia un numero natural $a(p)$, llamado la *aridad* de p.

(3) Ninguna palabra de $\mathcal{C}$ (resp. $\mathcal{F}$, $\mathcal{R}$) es subpalabra propia de otra palabra de $\mathcal{C}$ (resp. $\mathcal{F}$, $\mathcal{R}$).

Notese que los elementos de $\mathcal{C}$, $\mathcal{F}$ y $\mathcal{R}$ pueden ser palabras y no solo simbolos como en los casos de los tipos de estructuras conocidas. Mas adelante cuando definamos las *formulas de tipo* τ se entenderan las restricciones puestas en c. de (1) y en (3).

A los elementos de $\mathcal{C}$ (resp. $\mathcal{F}$, $\mathcal{R}$) los llamaremos *nombres de constante* (resp. *nombres de funcion, nombres de relacion*) *de tipo* τ. Para cada $n \in \mathbf{N}$, definamos

$$\begin{aligned}\mathcal{F}_n &= \{f \in \mathcal{F} : a(f) = n\} \\ \mathcal{R}_n &= \{r \in \mathcal{R} : a(r) = n\}\end{aligned}$$

Al tipo $(\emptyset, \emptyset, \{\leq\}, \{(\leq, 2)\})$ lo llamaremos el *tipo de los posets*. Al tipo $(\emptyset, \{\mathsf{s}, \mathsf{i}\}, \emptyset, \{(\mathsf{s}, 2), (\mathsf{i}, 2)\})$ lo llamaremos el *tipo de los reticulados terna*. Al tipo

$$(\{0, 1\}, \{\mathsf{s}, \mathsf{i}\}, \emptyset, \{(\mathsf{s}, 2), (\mathsf{i}, 2)\})$$

lo llamaremos el *tipo de los reticulados acotados*. Al tipo

$$(\{0, 1\}, \{\mathsf{s}, \mathsf{i}, c\}, \emptyset, \{(\mathsf{s}, 2), (\mathsf{i}, 2), (c, 1)\})$$

lo llamaremos el *tipo de los reticulados complementados*. Al tipo

$$(\emptyset, \{\mathsf{s}, \mathsf{i}\}, \{\leq\}, \{(\mathsf{s}, 2), (\mathsf{i}, 2), (\leq, 2)\})$$

lo llamaremos el *tipo de los reticulados cuaterna*. Al tipo $(\emptyset, \{M\}, \emptyset, \{(M, 3)\})$ lo llamaremos el *tipo de las median algebras*. Al tipo $(\emptyset, \emptyset, \{r\}, \{(r, 2)\})$ lo llamaremos el *tipo de los grafos*. Al tipo

$$(\emptyset, \emptyset, \{r, R\}, \{(r, 2), (R, 1)\})$$

lo llamaremos el *tipo de los grafos bicoloreados.*

Algunos ejemplos de tipos:

1. $(\{\text{uno}, \text{doli}\}, \{\text{MAS}, \text{P}\}, \{\text{Her}\}, a)$, con a dado por $a(\text{MAS}) = 4$, $a(\text{P}) = 1$ y $a(\text{Her}) = 3$.
2. $(\{0, 1\}, \{+, \times\}, \emptyset, a)$, con a dado por $a(+) = 2$ y $a(\times) = 2$.
3. $(\{\Box\}, \{\clubsuit\clubsuit, \text{Pic}\}, \{\rhd, \|\}, a)$, con a dado por $a(\clubsuit\clubsuit) = 6$, $a(\text{Pic}) = 1$, $a(\rhd) = 4$ y $a(\|) = 1$
4. $(\{\text{dod}, \text{dood}, \text{doood}, ...\}, \{\text{Fu}\}, \{\text{He}\}, a)$, con a dado por $a(\text{Fu}) = 1$ y $a(\text{He}) = 3$. Notese que este tipo tiene infinitos nombres de constante.

6.2 Estructuras de tipo τ

Ahora si estamos en condiciones de dar una definicion general de estructura. Daremos una definicion matematica de "Estructura de tipo τ". En virtud de nuestras estructuras conocidas uno podria intentar definir estructura de tipo τ como cierta n-upla pero esto trae problemas ya que en un tipo τ los nombres de $\mathcal{C} \cup \mathcal{F} \cup \mathcal{R}$ no tienen por que estar ordenados. De todas maneras la idea es muy similar y nos aproximaremos primero con ejemplos para entender mas facilmente el concepto.

Sea τ el tipo

$$(\{\text{uno}, \text{doli}\}, \{\text{MAS}, \text{P}\}, \{\text{Her}\}, \{(\text{MAS}, 4), (\text{P}, 1), (\text{Her}, 3)\})$$

Intuitivamente hablando, una estructura de tipo τ consiste en un conjunto no vacio A (que se llamara el universo de dicha estructura) junto con una interpretacion de cada uno de los nombres del conjunto $\{\text{uno}, \text{doli}, \text{MAS}, \text{P}, \text{Her}\}$. Esta interpretacion debe asignarle a

- uno un elemento de A
- doli un elemento de A
- MAS una operacion 4-aria sobre A
- P una operacion 1-aria sobre A
- Her una relacion 3-aria sobre A

Lo que debe quedar claro es que estos elementos, operaciones y relaciones pueden ser cualesquiera, es decir no deben cumplir nada en especial. Por ejemplo si tomamos $\mathbf{R}$ como universo podemos interpretar

- uno como el numero π
- doli como el numero 0
- MAS como la operacion

$$\begin{array}{ccc} \mathbf{R}^4 & \rightarrow & \mathbf{R} \\ (x,y,z,w) & \rightarrow & 2x+4y \end{array}$$

- P como la operacion

$$\begin{array}{ccc} \mathbf{R} & \rightarrow & \mathbf{R} \\ x & \rightarrow & 5^x \end{array}$$

- Her como la relacion

$$\{(x,y,z) \in \mathbf{R}^3 : x.y.z = 9\}$$

Analogamente, si τ es el tipo de los posets, es decir $\tau = (\emptyset, \emptyset, \{\leq\}, \{(\leq, 2)\})$, una estructura de tipo τ consistira de un conjunto no vacio A (que se llamara el universo de dicha estructura) junto con una interpretacion del simbolo $\leq$, la cual nos dira que relacion binaria sobre A denotara $\leq$. Pero esta relacion binaria puede ser cualquiera por lo cual habra muchas estructuras del tipo de los posets que no se corresponderan con posets. Solo aquellas en las que $\leq$ se interpreta como un orden parcial sobre su universo se corresponderan con los posets.

Ahora si daremos la definicion matematica de estructura de tipo τ. Sea τ un tipo. Una *estructura o modelo de tipo* τ sera un par $\mathbf{A} = (A, i)$ tal que:

(1) A es un conjunto no vacio

(2) i es una funcion con dominio $\mathcal{C} \cup \mathcal{F} \cup \mathcal{R}$, tal que:

   (a) $i(c)$ es un elemento de A, para cada $c \in \mathcal{C}$
   (b) $i(f)$ es una operacion n-aria sobre A, para cada $f \in \mathcal{F}_n$, $n \geq 1$
   (c) $i(r)$ es una relacion n-aria sobre A, para cada $r \in \mathcal{R}_n$, $n \geq 1$

Si $\mathbf{A} = (A, i)$ es una estructura de tipo τ, el conjunto A es llamado el *universo* de $\mathbf{A}$ y la funcion i es llamada la *funcion interpretacion* de $\mathbf{A}$. Si $s \in \mathcal{C} \cup \mathcal{F} \cup \mathcal{R}$, diremos que $i(s)$ es la interpretacion del simbolo s en $\mathbf{A}$. Algunos ejemplos:

(E1) Si τ es el tipo

$$(\{\text{uno}, \text{doli}\}, \{\text{MAS}, \text{P}\}, \{\text{Her}\}, \{(\text{MAS}, 4), (\text{P}, 1), (\text{Her}, 3)\})$$

entonces $(\mathbf{R}, i)$ es una estructura de tipo τ, si definimos i igual a la funcion con dominio $\{\text{uno}, \text{doli}, \text{MAS}, \text{P}, \text{Her}\}$ dada por

(a) $i(\text{uno}) = \pi$

(b) $i(\text{doli}) = 0$

(c) $i(\text{MAS})$ igual a la operacion

$$\begin{array}{rcl} \mathbf{R}^4 & \rightarrow & \mathbf{R} \\ (x,y,z,w) & \rightarrow & 2x+4y \end{array}$$

(d) $i(\text{P})$ igual a la operacion

$$\begin{array}{rcl} \mathbf{R} & \rightarrow & \mathbf{R} \\ x & \rightarrow & 5^x \end{array}$$

(e) $i(\text{Her}) = \{(x,y,z) \in \mathbf{R}^3 : x.y.z = 9\}$

(E2) Sea $\tau = (\emptyset, \emptyset, \{\leq\}, \{(\leq, 2)\})$. Notese que por definicion una estructura de tipo τ es un par (A,i) donde A es un conjunto no vacio y i es una funcion con dominio $\{\leq\}$ tal que $i(\leq)$ es una relacion binaria sobre A. Algunos ejemplos de estructuras de tipo τ:

(a) $(\{1,2,3\}, \{(\leq, \emptyset)\})$

(b) $(\{1,2,3\}, \{(\leq, \{2,3\} \times \{1\})\})$

(c) $(\{1, \{2\}, \emptyset\}, \{(\leq, \{(1, \{2\})\})\})$

(d) $(\mathbf{N}, i)$, con i dada por $i(\leq) = \{(1,2), (1000,1), (1,1)\}$

Notese que aunque τ es llamado el tipo de los posets, ninguna de las estructuras anteriores tiene mucho que ver con un poset. Consideremos ahora la estructura $(\mathbf{N}, i)$, donde i es la funcion con dominio igual a $\{\leq\}$ dada por

$$i(\leq) = \{(x,y) \in \mathbf{N}^2 : x|y\}$$

Notese que estrictamente hablando $(\mathbf{N}, i)$ no es un poset ya que i no es un orden parcial sobre $\mathbf{N}$ pero es claro que a nivel de informacion $(\mathbf{N}, i)$ y $(\mathbf{N}, |)$ son la misma cosa. O sea que aquellas estructuras de tipo τ en las cuales $\leq$ se interpreta como un orden parcial sobre el universo de la estructura son "esencialmente posets". Dejamos al lector dar una biyeccion entre el conjunto formado por todos los posets y un subconjunto del conjunto de todas las estructuras de tipo τ

(E3) Sea τ el tipo de los reticulados terna, es decir $\tau = (\emptyset, \{\mathsf{s}, \mathsf{i}\}, \emptyset, \{(\mathsf{s}, 2), (\mathsf{i}, 2)\})$. Entonces $(\mathbf{N}, i)$, donde $i = \{(\mathsf{s}, \max), (\mathsf{i}, \min)\}$, es una estructura de tipo τ. Notese que estrictamente hablando $(\mathbf{N}, i)$ no es un reticulado terna ya que es una 2-upla y los reticulados ternas son 3-uplas. Pero es claro que a nivel de informacion $(\mathbf{N}, i)$ y $(\mathbf{N}, \max, \min)$ son la misma cosa. Otras estructuras de tipo τ son por ejemplo:

(a) $(\mathbf{R}, \{(\mathsf{s}, +), (\mathsf{i}, \min)\})$

(b) $(\{0,1,2\},\{(\mathsf{s},f),(\mathsf{i},g)\}$ donde $f:\{0,1,2\}^2\to\{0,1,2\}$ es la funcion constantemente 1 y $g:\{0,1,2\}^2\to\{0,1,2\}$ es la funcion constantemente 2

Por supuesto, ninguna de las dos puede considerarse un reticulado terna ya que en ambas los simbolos s y i no se interpretan como las operaciones supremo e infimo provenientes de un orden parcial. Dejamos al lector dar una biyeccion entre el conjunto formado por todos los reticulados terna y un subconjunto del conjunto de todas las estructuras de tipo τ

(E4) Sea τ el tipo de los grafos bicoloreados, es decir $\tau=(\emptyset,\emptyset,\{r,R\},\{(r,2),(R,1)\})$. Entonces $(\{1,2\},i)$, con $i=\{(r,\{(1,2),(2,1)\}),(R,\{1\})\}$, es una estructura de tipo τ. Notese que

$$(\{1,2\},i(r),i(R))=(\{1,2\},\{(1,2),(2,1)\},\{1\})$$

es un grafo bicoloreado el cual esencialmente es lo mismo que la estructura $(\{1,2\},i)$ (a nivel de informacion). De todas maneras estrictamente hablando $(\{1,2\},i)$ no es un grafo bicoloreado. Otra estructura de tipo τ la cual es "esencialmente" un grafo bicoloreado es el par (ω,i), donde i es la funcion con dominio $\{r,R\}$ dada por

$$\begin{array}{rcl} i(r) & = & \{(x,x+1):x\in\omega\}\cup\{(x+1,x):x\in\omega\} \\ i(R) & = & \{x\in\omega:x\text{ es par}\} \end{array}$$

Tal como en los otros ejemplos vistos, hay estructuras de tipo τ las cuales no pueden considerarse grafos bicoloreados. Por ejemplo, la estructura $(\mathbf{N},\{(r,\{(1,2)\}),(R,\{3\})\})$. Dejamos al lector dar una biyeccion entre el conjunto formado por todos los grafos bicoloreados y un subconjunto del conjunto de todas las estructuras de tipo τ.

6.2.1 Independencia entre sintaxis y semantica

Notese que la definicion de tipo es muy libre en lo que respecta a que palabras componen los conjuntos $\mathcal{C}$, $\mathcal{F}$ y $\mathcal{R}$, es decir salvo por ciertas restricciones leves, ellas pueden ser cualquier palabra. Ademas no es necesario que las palabras de $\mathcal{C}\cup\mathcal{F}\cup\mathcal{R}$ se interpreten en la estructura de tipo τ (via la funcion i) como usualmente se interpretan en matematica. Algunos ejemplos:

1. $\tau=(\{\leq\},\emptyset,\emptyset,\emptyset)$ es un tipo y en las estructuras de tipo τ el simbolo $\leq$ se interpretara como un elemento del universo y no un orden parcial. Por ejemplo $(\{1,2,3\},\{(\leq,2)\})$ es una estructura de tipo τ.

2. $\tau' = (\emptyset, \emptyset, \{\leq\}, \{(\leq, 3)\})$ es un tipo pero en las estructuras de tipo τ' el simbolo $\leq$ se interpreta como una relacion 3-aria sobre el universo. Por ejemplo $(\mathbf{N}, i)$, con i dada por $i(\leq) = \{(x, y, z) \in \mathbf{N}^3 : x = y = z\}$, es una estructura de tipo τ'. En esta estructura el simbolo $\leq$ no se interpreta como un orden parcial sino como una relacion ternaria ya que en τ' el simbolo $\leq$ es un simbolo de relacion de aridad 3

3. $\tau'' = (\emptyset, \{1\}, \emptyset, \{(1, 3)\})$ es un tipo y en las estructuras de tipo τ'' el simbolo 1 se interpretara como una funcion 3-aria sobre el universo (tener cuidado al leer $(\emptyset, \{1\}, \emptyset, \{(1, 3)\})$ ya que en esta expresion 1 es el "numeral uno" y 3 es el numero tres). Por ejemplo si denotamos con f a la operacion

$$\begin{array}{ccc} \mathbf{Z}^3 & \rightarrow & \mathbf{Z} \\ (x, y, z) & \rightarrow & x + y + z \end{array}$$

entonces $(\mathbf{Z}, i)$, con i dada por $i(1) = f$, es una estructura de tipo τ''

Esta libertad en la definicion de tipo y tambien en la definicion de estructura de tipo τ (i.e. las estructuras interpretan a los nombres de $\mathcal{C} \cup \mathcal{F} \cup \mathcal{R}$ con total independencia de la fisonomia de las palabras de $\mathcal{C} \cup \mathcal{F} \cup \mathcal{R}$) es clave a la hora de fortalecer la separacion entre sintaxis y semantica, idea fundamental en el desarrollo de la logica. Para reforzar aun mas esta idea de independencia entre semantica y sintaxis veremos algunos ejemplos de conteo de estructuras. Antes un lema de conteo que nos sera de suma utilidad.

Lemma 174 *Se tiene que:*

(1) Dados A, B conjuntos finitos no vacios, hay $|B|^{|A|}$ funciones tales que su dominio es A y su imagen esta contenida en B

(2) si A es un conjunto cualquiera, entonces hay $2^{|A|}$ subconjuntos de A

Proof. (1) Supongamos $A = \{a_1, ..., a_n\}$, con $n = |A|$. Sea $Fu = \{f : D_f = A$ y $I_f \subseteq B\}$. Es facil ver que la siguiente funcion es biyectiva

$$\begin{array}{ccc} Fu & \rightarrow & B^n \\ f & \rightarrow & (f(a_1), ..., f(a_n)) \end{array}$$

(2) Ya que los subconjuntos de A estan en correspondencia biunivoca con las funciones de A en $\{0, 1\}$ (por que?) podemos aplicar (1) $\blacksquare$

Daremos a continuacion algunos ejemplos de conteo de estructuras. Sea

$$\tau = (\emptyset, \emptyset, \{\leq\}, \{(\leq, 2)\})$$

Nos interesa saber cuantas estructuras de tipo τ hay que tengan al conjunto $\{1, 2, 3\}$ como universo. Una estructura de tipo τ con universo $\{1, 2, 3\}$ es un par $(\{1, 2, 3\}, i)$ donde i es una funcion tal que su dominio es $\{\leq\}$ y tal que

- $i(\leq)$ es una relacion 2-aria sobre $\{1,2,3\}$, es decir es un subconjunto de $\{1,2,3\}^2$

O sea que una estructura de tipo τ con universo $\{1,2,3\}$ es un par de la forma

$$(\{1,2,3\},\{(\leq,S)\})$$

donde S es cualquier subconjunto de $\{1,2,3\}^2$. Ya que, por el lema anterior, hay 2^9 subconjuntos del conjunto $\{1,2,3\}^2$, tenemos que hay exactamente 2^9 estructuras de tipo τ cuyo universo es $\{1,2,3\}$. Notese que, estrictamente hablando, ninguna de estas estructuras es un poset. Sin embargo aquellas en las cuales S es un orden parcial sobre $\{1,2,3\}$ pueden considerarse como posets ya que esencialmente estan determinadas por un orden parcial.

Otro ejemplo, tomemos

$$\tau = (\{\mathrm{un},\mathrm{do}\},\{\mathrm{MAS},\mathrm{P}\},\{\mathrm{Her}\},\{(\mathrm{MAS},4),(\mathrm{P},1),(\mathrm{Her},3)\}$$

Nos interesa saber cuantas estructuras de tipo τ hay que tengan al conjunto $\{1,2,3\}$ como universo. Una estructura de tipo τ con universo $\{1,2,3\}$ es un par $(\{1,2,3\},i)$ donde i es una funcion tal que su dominio es $\{\mathrm{un},\mathrm{do},\mathrm{MAS},\mathrm{P},\mathrm{Her}\}$ y tal que

1. $i(\mathrm{un})$ y $i(\mathrm{do})$ pertenecen a $\{1,2,3\}$
2. $i(\mathrm{MAS})$ es una operacion 4-aria sobre $\{1,2,3\}$
3. $i(\mathrm{P})$ es una operacion 1-aria sobre $\{1,2,3\}$
4. $i(\mathrm{Her})$ es una relacion 3-aria sobre $\{1,2,3\}$, es decir es un subconjunto de $\{1,2,3\}^3$

Notese que hay

1. 3 posibilidades para $i(\mathrm{un})$
2. 3 posibilidades para $i(\mathrm{do})$
3. $3^{(3^4)}$ posibilidades para $i(\mathrm{MAS})$ (por (1) del lema anterior con $A=\{1,2,3\}^4$ y $B=\{1,2,3\}$)
4. 3^3 posibilidades para $i(\mathrm{P})$ (por (1) del lema anterior con $A=\{1,2,3\}$ y $B=\{1,2,3\}$)
5. $2^{(3^3)}$ posibilidades para $i(\mathrm{Her})$ (por (2) del lema anterior con $A=\{1,2,3\}^3$)

O sea que hay exactamente $3.3.3^{(3^4)}.3^3.2^{(3^3)}$ estructuras de tipo τ que tienen al conjunto $\{1,2,3\}$ como universo.

6.3 Un poco de arrogancia

Hemos dado, via las definiciones de *tipo* y de *estructura de tipo* τ, un modelo matematico preciso del concepto intuitivo de estructura que veniamos acuñando en las guias anteriores. Esto es un salto importante ya que ahora tenemos una definicion matematica de lo que es una estructura en general y no solo un puñado de definiciones matematicas de ciertas estructuras particulares. Hemos encontrado la esencia del concepto intuitivo de estructura que veniamos acuñando con casos particulares en las primeras guias. La modelizacion es bastante sofisticada al punto que ninguna de las estructuras concretas antes estudiadas es estrictamente hablando una estructura de tipo τ, aunque cada tipo de estructura concreta estudiada tiene su "version" dentro de esta definicion general de estructura de tipo τ, la cual es esencialmente el mismo objeto. Por ejemplo, para el tipo de los reticulados complementados

$$\tau = (\{0,1\}, \{\mathsf{s}, \mathsf{i}, c\}, \emptyset, \{(\mathsf{s},2), (\mathsf{i},2), (c,1)\})$$

las estructuras de tipo τ que modelizan a los reticulados complementados son precisamente aquellas estructuras (A, i) tales que

$$(A, i(\mathsf{s}), i(\mathsf{i}), i(c), i(0), i(1))$$

es un reticulado complementado. Obviamente estas estructuras no son estrictamente hablando reticulados complementados, pero esencialmente son la misma cosa.

La utilidad de este nuevo concepto general de estructura ira quedando clara a medida que avancemos. Cabe destacar que este concepto general de estructura no solo ha sido clave en el desarrollo de la logica matematica sino que tambien ha sido crusial en el desarrollo de la informatica teorica, mas precisamente en el area de especificaciones algebraicas, ya que la versatilidad del concepto de estructura eterogenea ha permitido crear una teoria de amplio alcance y modelizacion de la idea de la especificacion de tipos abstractos de datos.

6.4 Formulas elementales de tipo τ

Recordemos que cada una de las estructuras consideradas en la Guia 6 tiene su tipo asociado. Es decir:

$$\begin{array}{rcl}
\text{Tipo de los posets} & = & (\emptyset, \emptyset, \{\leq\}, \{(\leq, 2)\}) \\
\text{Tipo de los ret. ternas} & = & (\emptyset, \{\mathsf{s}, \mathsf{i}\}, \emptyset, \{(\mathsf{s},2), (\mathsf{i},2)\}) \\
\text{Tipo de los ret. acotados} & = & (\{0,1\}, \{\mathsf{s}, \mathsf{i}\}, \emptyset, \{(\mathsf{s},2), (\mathsf{i},2)\}) \\
\text{Tipo de los ret. comp.} & = & (\{0,1\}, \{\mathsf{s}, \mathsf{i}, c\}, \emptyset, \{(\mathsf{s},2), (\mathsf{i},2), (c,1)\}) \\
\text{Tipo de los ret. cuaternas} & = & (\emptyset, \{\mathsf{s}, \mathsf{i}\}, \{\leq\}, \{(\mathsf{s},2), (\mathsf{i},2), (\leq,2)\}) \\
\text{Tipo de las median algebras} & = & (\emptyset, \{M\}, \emptyset, \{(M,3)\}) \\
\text{Tipo de los grafos} & = & (\emptyset, \emptyset, \{r\}, \{(r,2)\}) \\
\text{Tipo de los grafos bicoloreados} & = & (\emptyset, \emptyset, \{r, R\}, \{(r,2), (R,1)\})
\end{array}$$

Notese que en cada uno de los casos anteriores los simbolos de $\mathcal{C} \cup \mathcal{F} \cup \mathcal{R}$ son los que se usan (junto con los simbolos logicos, las variables y los simbolos de elementos fijos) para formar sus correspondientes formulas elementales. Es decir, lo particular de las formulas elementales de cada tipo de estructura estaba dado por los correspondientes simbolos de $\mathcal{C} \cup \mathcal{F} \cup \mathcal{R}$. Esto nos permite generalizar nuestro concepto intuitivo de formula elemental para el caso de cualquier tipo τ de estructuras.

Si $\tau = (\mathcal{C}, \mathcal{F}, \mathcal{R}, a)$ es un tipo, las *formulas elementales de tipo* τ se construyen en forma finitaria usando los nombres de $\mathcal{C} \cup \mathcal{F} \cup \mathcal{R}$ junto con los simbolos de la siguiente lista:

$\forall \; \exists \; \neg \; \vee \; \wedge \; \rightarrow \; \leftrightarrow \; (\;) \; , \; =$

Variables: $x, y, z, w, ...$

Nombres para elementos fijos: $a, b, c, d, ...$

La manera en la que se construyen las formulas elementales de tipo τ a partir de los simbolos anteriores y de las palabras de $\mathcal{C} \cup \mathcal{F} \cup \mathcal{R}$ es esencialmente la misma que usamos en la Seccion 5 para construir las formulas elementales de los distintos tipos de estructuras clasicas. Mostraremos esto con varios ejemplos asi el lector queda con una idea clara del concepto.

Por supuesto el concepto de formula elemental de tipo τ no es un concepto definido en forma precisa sino mas bien una idea basada en ciertos ejemplos de la vida real de los matematicos.

(E1) Si τ es el tipo

$$(\{\mathrm{un}, 0\}, \{\mathrm{MAS}, \mathrm{P}\}, \{\mathrm{Her}, \mathrm{Verde}\}, \{(\mathrm{MAS}, 4), (\mathrm{P}, 1), (\mathrm{Her}, 3), (\mathrm{Verde}, 1)\})$$

entonces las siguientes son formulas elementales de tipo τ:

(a) $\mathrm{Her}(x, y, z)$
(b) $\mathrm{Verde}(x)$
(c) $\mathrm{Verde}(\mathrm{MAS}(a, b, \mathrm{un}, z))$
(d) $\mathrm{Her}(0, \mathrm{MAS}(a, b, \mathrm{un}, z), \mathrm{P}(\mathrm{P}(z)))$
(e) $(\mathrm{un} = \mathrm{do})$
(f) $(\mathrm{Verde}(\mathrm{MAS}(a, b, \mathrm{un}, z)) \wedge (\mathrm{un} = \mathrm{do}))$
(g) $(\mathrm{MAS}(a, b, \mathrm{un}, z) = b)$
(h) $(\mathrm{MAS}(a, b, \mathrm{un}, \mathrm{P}(z)) = \mathrm{P}(\mathrm{P}(\mathrm{P}(z))))$
(i) $\exists z (\mathrm{MAS}(a, b, \mathrm{un}, z) = b)$
(j) $\forall x \forall y \mathrm{Her}(0, y, \mathrm{P}(\mathrm{P}(x)))$
(k) $\forall y \; ((\mathrm{P}(\mathrm{P}(z)) = x) \rightarrow \exists z \; (\mathrm{Verde}(z) \wedge \mathrm{Her}(x, y, z)))$

Por supuesto las aridades de los nombres de $\mathcal{F} \cup \mathcal{R}$ son importantes y deben ser respetadas. Por ejemplo

$$(\mathrm{P}(x,y) = x) \qquad \mathrm{Her}(x,y) \qquad \mathrm{Verde}(x,y)$$

no son formulas elementales de tipo τ.

(E2) Si τ es el tipo

$$(\{0,1\},\{+,\triangle\},\{\leq,r\},\{(+,2),(\triangle,5),(\leq,2),(r,2)\})$$

entonces las siguientes son formulas elementales de tipo τ:

(a) $r(x,z)$
(b) $\leq(x,y)$
(c) $\leq(\triangle(x,y,z,0,0),+(x,x))$
(d) $(+(a,b) = \triangle(x,y,z,0,0))$
(e) $(\triangle(x,y,z,0,0) = \triangle(1,1,0,x,z))$
(f) $(+(\triangle(x,y,z,0,0),z) = 1)$
(g) $\neg r(x,+(a,+(a,b)))$
(h) $\neg\forall y(+(x,y) = x)$
(i) $\exists z\forall x\ (r(x,+(z,z) \wedge \neg\leq(x,z))$
(j) $\forall x\forall y\forall z\ ((r(x,y) \wedge r(y,z)) \rightarrow r(x,z))$

Notese que hay algunas pequeñas diferencias con las formulas elementales de las estructuras clasicas ya que aqui respondemos a un formato mas general. Por ejemplo hemos escrito $\leq(x,y)$ en lugar de $x \leq y$ y $+(x,y)$ en lugar de $x + y$. Esto es a los fines de homogeneisar la escritura y no hacer un uso distinto para las operaciones binarias y las relaciones binarias.

Por supuesto las aridades de los nombres de $\mathcal{F} \cup \mathcal{R}$ son importantes y deben ser respetadas. Por ejemplo

$$(+(x,y,z) = x) \qquad r(x,y,z) \qquad \leq(x,y,z)$$

no son formulas elementales de tipo τ.

(E3) Si τ es el tipo

$$(\{\mathrm{er}\},\{+\},\{\leq\},\{(+,4),(\leq,5)\})$$

entonces las siguientes son formulas elementales de tipo τ:

(a) $\leq(x,y,\mathrm{er},\mathrm{er},\mathrm{er})$
(b) $\leq(+(x,y,z,\mathrm{er}),+(x,x,\mathrm{er},x),a,b,z)$
(c) $\exists z(+(x,z,x,+(x,x,x,x)) = z)$

(E4) Si τ es el tipo

$$(\{\text{er}\}, \{\leq\}, \{+\}, \{(\leq, 3), (+, 2)\})$$

entonces las siguientes son formulas elementales de tipo τ:

(a) $(\leq(x, y, \text{er}) = x)$

(b) $+(z, \text{er})$

(c) $\exists z \neg +(z, \text{er})$

(aqui hay que tener en cuenta que $\leq$ es un nombre de funcion de aridad 3 y que $+$ es un nombre de relacion de aridad 2, lo cual es inusual pero perfectamente posible en nuestra muy general definicion de tipo)

(E5) Si τ es el tipo

$$(\{\leq\}, \{+\}, \emptyset, \{(+, 3)\})$$

entonces las siguientes son formulas elementales de tipo τ:

(a) $(\leq = x)$

(b) $(+(z, \leq, a) = \leq)$

(c) $(+(+(z, \leq, \leq), x, a) = b)$

(aqui hay que tener en cuenta que $\leq$ es un nombre de constante, lo cual es inusual pero perfectamente posible en nuestra muy general definicion de tipo)

Para que una formula elemental de tipo τ se vuelva verdadera o falsa tenemos que tener una estructura (A, i) de tipo τ y ademas asignarles valores concretos de A a las variables libres y a los nombres de elementos fijos que figuran en dicha formula. Cuando la formula no tiene variables libres diremos que es una *sentencia elemental de tipo* τ. Notese que en tal caso sera verdadera o falsa en una estructura dada dependiendo solo de los valores que tomen los nombres para elementos fijos que ocurren en ella. Tambien cabe destacar que los cuantificadores siempre ranguean sobre A, es decir $\forall x$ se interpretara como $\forall x \in A$ y $\exists x$ se interpretara como $\exists x \in A$. La diferencia entre las variables y los nombres de elementos fijos es que si bien ambos pueden variar su valor los nombres de elementos fijos suelen denotar un valor fijo de la estructura durante todo un desarrollo o demostracion. Tampoco se cuantificaran los nombres de elementos fijos, es decir solo cuantificamos variables (por ejemplo, $\forall a(a = x)$ no es una formula elemental de tipo τ, cualquiera sea el tipo τ).

Daremos algunos ejemplos para reafirmar la idea intuitiva de cuando una formula elemental es verdadera en una estructura dada para una asignacion de valores de sus variables libres y de sus nombres de elementos fijos:

(E1) Sea τ el tipo

$$(\{\text{un}, 0\}, \{\text{MAS}, \text{P}\}, \{\text{Her}, \text{Verde}\}, \{(\text{MAS}, 4), (\text{P}, 1), (\text{Her}, 3), (\text{Verde}, 1)\})$$

y sea (A, i) la estructura de tipo τ dada por:

- $A = \mathbf{R}$, $i(\text{un}) = \pi$, $i(0) = 0$ (ojo que aqui el primer cero es un simbolo y el segundo un numero real!)

-

$$\begin{array}{rcl} i(\text{MAS}) : \mathbf{R}^4 & \to & \mathbf{R} \\ (x, y, z, w) & \to & x.y \end{array}$$

$$\begin{array}{rcl} i(\text{P}) : \mathbf{R} & \to & \mathbf{R} \\ x & \to & x^2 \end{array}$$

$$\begin{array}{rcl} i(\text{Her}) & = & \{(x, y, z) \in \mathbf{R}^3 : x.y.z = 9\} \\ i(\text{Verde}) & = & \mathbf{Q} \end{array}$$

Entonces:

(a) la formula $\text{Her}(x, y, z)$ es verdadera en (A, i) cuando le asignamos a x el valor 9, a y el valor 1 y a z el valor 1

(b) $\text{Verde}(x)$ es falsa en (A, i) cuando le asignamos a x el valor $\sqrt{2}$

(c) $\text{Verde}(\text{MAS}(a, b, \text{un}, z))$ es verdadera en (A, i) cuando le asignamos a a el valor $\sqrt{2}$, a b el valor $\sqrt{2}$ y a z el valor 16 (o cualquier otro valor)

(d) la formula $\exists y \exists z\ \text{Her}(a, y, z))$ es una sentencia ya que no tiene variables libres y es veradera en (A, i) cuando a a le asignamos un valor no nulo

(e) la formula $\exists y \exists z\ \text{Her}(x, y, z))$ es una formula y es veradera en (A, i) cuando a x le asignamos un valor no nulo

(f) la formula $\forall x\ (\neg(x = 0) \to \exists y \exists z\ \text{Her}(x, y, z))$ es una sentencia ya que no tiene variables libres y es veradera en (A, i)

(g) la formula $\forall x \forall y\ ((\text{Verde}(x) \wedge \text{Verde}(y)) \to \text{Verde}(\text{MAS}(x, y, \text{un}, z)))$ es verdadera en (A, i) independientemente de que valor le asignemos a z, ya que el producto de racionales es racional

(h) la formula $\exists y(\text{MAS}(z, z, y, \text{un}) = \text{P}(z))$ es veradera en (A, i) cualquiera sea el valor que le asignemos a z

(i) **Error frecuente:** En la estructura anterior hay varios elementos que tienen su notacion clasica en la matematica, por ejemplo, con la letra griega π denotamos la cantidad de veces que entra el diametro en la circunferencia o con el numeral 3 denotamos al numero entero tres. Esto no debe confundirnos y pensar que por ejemplo las palabras

$$\neg\text{Verde}(\pi) \qquad\qquad \exists y \text{Her}(3, 3, y)$$

son formulas elementales de tipo τ (aunque es claro que son verdaderas en la estructura (A, i))

(E2) Sea τ el tipo

$$(\{\text{er}\},\{+\},\{\leq\},\{(+,4),(\leq,5)\})$$

y sea (A,i) la estructura de tipo τ dada por:

- $A=\{1,2,3,4,5\}$, $i(\text{er})=4$
-
$$\begin{array}{rcl} i(+):A^4 & \rightarrow & A \\ (x,y,z,w) & \rightarrow & \max\{x,y,z,w\} \end{array}$$
$$i(\leq)=\{(x,y,z,u,v)\in A^5 : x+y+z+u+v\geq 17\}$$

Entonces:

(a) $\leq(\text{er},\text{er},\text{er},\text{er},\text{er})$ es una sentencia verdadera en (A,i)

(b) $\leq(x,y,\text{er},\text{er},\text{er})$ es verdadera en (A,i) cuando le asignamos a las variables x e y valores que sumados den al menos 5

(c) $\forall x\exists y\leq(x,x,x,x,y)$ es una sentencia la cual es falsa en (A,i), ya que la formula $\exists y\leq(x,x,x,x,y)$ es falsa en (A,i) cuando le asignamos a x el valor 1

(d) la sentencia $\forall x\exists z\leq(x,x,x,x,+(x,x,x,z))$ es verdadera en (A,i)

(E3) Sea τ el tipo

$$(\{\text{epa}\},\{\leq,r\},\emptyset,\{(\leq,1),(r,1)\})$$

y sea (A,i) la estructura de tipo τ dada por:

- $A=\omega$, $i(\text{epa})=71$
-
$$\begin{array}{rcl} i(\leq):\omega & \rightarrow & \omega \\ x & \rightarrow & x^2 \end{array}$$
$$\begin{array}{rcl} i(r):\omega & \rightarrow & \omega \\ x & \rightarrow & \lfloor\sqrt{x}\rfloor \end{array}$$

(Notese que aqui contrario al uso estandard en la matematica, el simbolo $\leq$ se interpreta como una funcion.) Entonces:

(a) $(\leq(\text{epa})=x)$ es veradera en (A,i) cuando le asignamos a la variable x el valor 71^2 y falsa en caso contrario

(b) la sentencia $\exists z(\leq(z)=x)$ es verdadera en (A,i) cuando le asignamos a x el valor 16

(c) la sentencia $\forall x\ (r(\leq(x))=x)$ es verdadera en (A,i)

(d) la sentencia $\exists x\ \neg(\leq(r(x))=x)$ es verdadera en (A,i)

6.5 Teorias elementales y pruebas elementales

Tal como vimos en la Seccion 5, el concepto de prueba elemental dependia del tipo de estructura en cuestion y ademas de tener fijado un conjunto de sentencas elementales que llamabamos axiomas y eran el punto de partida de dichas pruebas. Cabe destacar que dichos axiomas eran sentencias elementales sin nombres de elementos fijos ya que estos se usaban solo en las pruebas elementales para denotar hipoteticos elementos dentro del argumento de la prueba misma. Ademas cuando haciamos una prueba elemental teniamos en mente una estructura generica de la cual solo sabiamos que satisfacia los axiomas, es decir solo podiamos usar la informacion particular que dichos axiomas nos proveian y pasos elementales obvios de los cuales nadie dudaria. Esto nos inspira a hacer las siguientes dos definiciones.

Una *teoria elemental* sera un par (Σ, τ) tal que τ es un tipo cualquiera y Σ es un conjunto de sentencias elementales de tipo τ, las cuales no tienen nombres de elementos fijos. Un *modelo de* (Σ, τ) sera una estructura de tipo τ la cual haga verdaderos a todos los elementos de Σ. Veamos algunos ejemplos:

(E1) La *teoria elemental de los posets* es el par (Σ, τ), donde $\tau = (\emptyset, \emptyset, \{\leq\}, \{(\leq, 2)\})$ y Σ es el conjunto formado por las siguientes tres sentencias elementales de tipo τ:

(a) $\forall x \leq(x, x)$

(b) $\forall x \forall y \forall z\ ((\leq(x, y) \wedge \leq(y, z)) \rightarrow \leq(x, z))$

(c) $\forall x \forall y((\leq(x, y) \wedge \leq(y, x)) \rightarrow x = y)$

Notese que los modelos de esta teoria elemental son exactamente aquellas estructuras de tipo τ las cuales son "esencialmente" posets.

(E2) La *teoria elemental de los reticulados terna* es el par (Σ, τ), donde $\tau = (\emptyset, \{\mathsf{s}, \mathsf{i}\}, \emptyset, \{(\mathsf{s}, 2), (\mathsf{i}, 2)\})$ y Σ es el conjunto formado por las siguientes sentencias elementales de tipo τ:

(a) $\forall x \forall y\ (\mathsf{s}(x, x) = x)$

(b) $\forall x \forall y\ (\mathsf{i}(x, x) = x)$

(c) $\forall x \forall y\ (\mathsf{s}(x, y) = \mathsf{s}(y, x))$

(d) $\forall x \forall y\ (\mathsf{i}(x, y) = \mathsf{i}(y, x))$

(e) $\forall x \forall y \forall z\ (\mathsf{s}(\mathsf{s}(x, y), z) = \mathsf{s}(x, \mathsf{s}(y, z)))$

(f) $\forall x \forall y \forall z\ (\mathsf{i}(\mathsf{i}(x, y), z) = \mathsf{i}(x, \mathsf{i}(y, z)))$

(g) $\forall x \forall y\ \mathsf{s}(x, \mathsf{i}(x, y)) = x)$

(h) $\forall x \forall y\ \mathsf{i}(x, \mathsf{s}(x, y)) = x)$

Notese que los modelos de esta teoria elemental son exactamente aquellas estructuras de tipo τ las cuales son "esencialmente" reticulados terna.

(E3) La *teoria elemental de los reticulados cuaterna* es el par (Σ, τ), donde $\tau = (\emptyset, \{\mathsf{s}, \mathsf{i}\}, \{\leq\}, \{(\mathsf{s}, 2), (\mathsf{i}, 2), (\leq, 2)\})$ y Σ es el conjunto formado por las siguientes sentencias elementales de tipo τ:

(a) $\mathrm{A}_{\leq R} = \forall x \leq(x, x)$
(b) $\mathrm{A}_{\leq T} = \forall x \forall y \forall z\ ((\leq(x, y) \wedge \leq(y, z)) \rightarrow \leq(x, z))$
(c) $\mathrm{A}_{\leq A} = \forall x \forall y\ ((\leq(x, y) \wedge \leq(y, x)) \rightarrow x = y)$
(d) $\mathrm{A}_{\mathsf{s}esC} = \forall x \forall y\ (\leq(x, \mathsf{s}(x, y)) \wedge \leq(y, \mathsf{s}(x, y)))$
(e) $\mathrm{A}_{\mathsf{s}\leq C} = \forall x \forall y \forall z\ ((\leq(x, z) \wedge \leq(y, z)) \rightarrow \leq(\mathsf{s}(x, y), z))$
(f) $\mathrm{A}_{\mathsf{i}esC} = \forall x \forall y\ (\leq(\mathsf{i}(x, y), x) \wedge \leq(\mathsf{i}(x, y), y))$
(g) $\mathrm{A}_{\mathsf{i}\geq C} = \forall x \forall y \forall z\ ((\leq(z, x) \wedge \leq(z, y)) \rightarrow \leq(z, \mathsf{i}(x, y)))$

Notese que los modelos de esta teoria elemental son exactamente aquellas estructuras de tipo τ las cuales son "esencialmente" reticulados cuaterna.

(E4) La *teoria elemental de los grafos* es el par (Σ, τ), donde $\tau = (\emptyset, \emptyset, \{r\}, \{(r, 2)\})$ y Σ es el conjunto formado por la siguiente sentencia elemental de tipo τ:

(a) $\forall x \forall y (r(x, y) \rightarrow r(y, x))$

Notese que los modelos de esta teoria elemental son exactamente aquellas estructuras de tipo τ las cuales son "esencialmente" grafos.

(E5) La *teoria elemental de los grafos bicoloreados* es el par (Σ, τ), donde $\tau = (\emptyset, \emptyset, \{r, R\}, \{(r, 2), (R, 1)\})$ y Σ es el conjunto formado por las siguientes sentencias elementales de tipo τ:

(a) $\forall x \forall y (r(x, y) \rightarrow r(y, x))$
(b) $\forall x \forall y (r(x, y) \rightarrow (R(x) \leftrightarrow \neg R(y)))$

Notese que los modelos de esta teoria elemental son exactamente aquellas estructuras de tipo τ las cuales son "esencialmente" grafos bicoloreados.

Es muy importante notar que una teoria elemental (Σ, τ) es en algun sentido un objeto esencialmente sintactico ya que Σ, $\mathcal{C}$, $\mathcal{F}$ y $\mathcal{R}$ son conjuntos de palabras. Los modelos de (Σ, τ) constituyen la semantica de la teoria.

Las anteriores son las teorias elementales que se corresponden con los tipos de estructuras consideradas en la Seccion 5 pero nuestra definicion de teoria elemental es muy general y nos permite considerar una gran diversidad de teorias. Veamos algunos ejemplos de teorias elementales interesantes y no consideradas en la Seccion 5:

(E6) Consideremos la teoria elemental (Σ, τ), donde $\tau = (\{\mathrm{ex}\}, \{\mathrm{F}\}, \emptyset, \{(\mathrm{F}, 1)\})$ y Σ es el conjunto formado por las siguientes dos sentencias elementales:

(a) $\forall x \forall y\ (\neg(x \equiv y) \rightarrow \neg(\mathrm{F}(x) \equiv \mathrm{F}(y)))$

(b) $\forall x\ \neg(\mathrm{F}(x) \equiv \mathrm{ex})$

Notese que una estructura $\mathbf{A} = (A, i)$ de tipo τ es un modelo de (Σ, τ) si y solo si $i(\mathrm{F})$ es inyectiva y $i(\mathrm{ex}) \notin \mathrm{Im}(i(\mathrm{F}))$. Esto obviamente nos dice que el universo de cada modelo de esta teoria es infinito. Un modelo de la teoria es por ejemplo $(\omega, \{(\mathrm{ex}, 0), (\mathrm{F}, Suc)\})$

(E7) Sea $\tau = (\emptyset, \{\times\}, \{\mathrm{Com}\}, \{(\times, 2), (\mathrm{Com}, 1)\})$ y sea Σ el conjunto formado por las siguientes sentencias elementales de tipo τ:

(a) $\forall x \forall y \forall z\ (\times(\times(x, y), z) \equiv \times(x, \times(y, z)))$

(b) $\forall z\ (\mathrm{Com}(z) \rightarrow \forall x\ (\times(x, z) \equiv \times(z, x)))$

(c) $\forall x \exists z\ (x \equiv \times(z, z) \wedge \mathrm{Com}(z))$

Supongamos $\mathbf{A} = (A, i)$ es un modelo de la teoria (Σ, τ). Notese que el primer axioma nos dice que $i(\times)$ es una operacion binaria asociativa, esto se ve mas facilmente si escribimos dicho axioma con la notacion mas usual para operaciones:

$$\forall x \forall y \forall z\ (x \times y) \times z \equiv x \times (y \times z)$$

El segundo axioma nos dice que si $a \in i(\mathrm{Com})$, entonces $a\ i(\times)\ b = b\ i(\times)\ a$, cualesquiera sea $b \in A$. O sea nos dice que los elementos de $i(\mathrm{Com})$ conmutan con todos los otros elementos relativo a la operacion $i(\times)$. El tercer axioma nos dice que cualquiera sea $a \in A$, debe haber un $b \in i(\mathrm{Com})$ tal que $b\ i(\times)\ b = a$. En algun sentido nos dice que todo elemento de A tiene en el conjunto $i(\mathrm{Com})$ una "raiz cuadrada" relativo a la operacion $i(\times)$. Ejemplos de modelos de esta teoria son:

(a) $(\{r \in \mathbf{R} : r \geq 0\}, i)$, con $i(\times) =$ operacion producto usual de $\mathbf{R}$ restringida a $\{r \in \mathbf{R} : r \geq 0\}^2$ y $i(\mathrm{Com}) = \{r \in \mathbf{R} : r \geq 0\}$

(b) $(\mathbf{R}, i)$, con $i(\times) = \max$ y $i(\mathrm{Com}) = \mathbf{R}$

(c) $(\mathbf{R}, i)$, con $i(\times) = \min$ y $i(\mathrm{Com}) = \mathbf{R}$

(d) $(\mathcal{P}(\{1, 2, 3\}), i)$, con $i(\times) = \cup$ y $i(\mathrm{Com}) = \mathcal{P}(\{1, 2, 3\})$

(E8) La *teoria elemental de los reticulados cuaterna distributivos* es el par (Σ, τ), donde $\tau = (\emptyset, \{\mathsf{s}, \mathsf{i}\}, \{\leq\}, \{(\mathsf{s}, 2), (\mathsf{i}, 2), (\leq, 2)\})$ y Σ es el conjunto formado por los axiomas de la teoria elemental de los reticulados cuaterna junto con el axioma

(a) $\forall x \forall y \forall z\ (\mathsf{i}(x, \mathsf{s}(y, z)) \equiv \mathsf{s}(\mathsf{i}(x, y), \mathsf{i}(x, z)))$

Notese que los modelos de esta teoria elemental son exactamente aquellas estructuras de tipo τ las cuales son "esencialmente" reticulados cuaterna distributivos

(E9) La *teoria elemental de los reticulados terna distributivos* es el par (Σ, τ), donde $\tau = (\emptyset, \{\mathsf{s}, \mathsf{i}\}, \emptyset, \{(\mathsf{s}, 2), (\mathsf{i}, 2)\})$ y Σ es el conjunto formado por los axiomas de la teoria elemental de los reticulados terna junto con el axioma

(a) $\forall x \forall y \forall z\ (\mathsf{i}(x, \mathsf{s}(y, z)) = \mathsf{s}(\mathsf{i}(x, y), \mathsf{i}(x, z)))$

Notese que los modelos de esta teoria elemental son exactamente aquellas estructuras de tipo τ las cuales son "esencialmente" reticulados terna distributivos

6.5.1 Pruebas elementales

Podemos generalizar el concepto de prueba elemental, introducido en la Seccion 5, a cualquier teoria elemental. Dada una teoria elemental (Σ, τ) y una sentencia elemental φ la cual no posea nombres de constantes auxiliares, una *prueba elemental de* φ *en* (Σ, τ) sera una prueba de φ que posea las siguientes caracteristicas:

- salvo por ciertas aclaraciones simples y concretas en castellano la prueba se escribe usando solo sentencias elementales de tipo τ
- cada paso de la demostracion debe ser obvio y solido
- cuando el matematico comiensa la prueba tiene en mente una estructura generica de tipo τ que satisface los axiomas de la teoria (i.e. es un modelo de la teoria) y esa es la unica particularidad que supone de dicho modelo generico al comensar la prueba. Es decir las pruebas elementales siempre son formas solidas de justificar que *cualquier* estructura que satisfaga los axiomas tambien satisfacera la sentencia probada

Por supuesto el concepto de prueba elemental en una teoria (Σ, τ) no es un concepto definido en forma precisa sino mas bien una idea basada en ciertos ejemplos de la vida real de los matematicos.

Veamos algunos ejemplos:

(E10) Consideremos la teoria elemental del ejemplo (E6). Sea

$$\varphi = \exists x \exists y \exists z\ (\neg(x = y) \wedge \neg(x = z) \wedge \neg(y = z))$$

(φ dice que el universo tiene al menos tres elementos.) Tenemos la siguiente:

Prueba elemental de φ en (Σ, τ): Por el segundo axioma tenemos que $\neg(\mathrm{F}(\mathrm{ex}) = \mathrm{ex})$. Obviamente entonces tenemos que

(1) $\neg(\mathrm{ex} = \mathrm{F}(\mathrm{ex}))$

Por el segundo axioma tambien tenemos que $\neg(\text{F}(\text{F}(\text{ex})) = \text{ex})$ por lo que

(2) $\neg(\text{ex} = \text{F}(\text{F}(\text{ex})))$

Ya que se da (2), el primer axioma nos dice que

(3) $\neg(\text{F}(\text{ex}) = \text{F}(\text{F}(\text{ex})))$

Poniendo (1), (2) y (3) juntos tenemos que

$$\neg(\text{ex} = \text{F}(\text{ex})) \wedge \neg(\text{ex} = \text{F}(\text{F}(\text{ex}))) \wedge \neg(\text{F}(\text{ex}) = \text{F}(\text{F}(\text{ex})))$$

de lo cual es obvio que vale φ.

(E11) Consideremos la teoria elemental del ejemplo (E7). A continuacion daremos una prueba elemental de $\varphi = \forall x \forall y\ (\times(x, y) = \times(y, x))$ en la teoria (Σ, τ). Para facilitar la lectura usaremos la notacion clasica para operaciones binarias, es decir escribiremos $x \times y$ en lugar de $\times(y, x)$, etc.

Prueba elemental de φ en (Σ, τ): Sean $a, b \in A$, fijos pero arbitrarios. Por el tercer axioma tenemos que

1. $\exists z\ (a = z \times z \wedge \text{Com}(z))$

Sea c tal que

2. $a = c \times c \wedge \text{Com}(c)$

Nuevamente, por el tercer axioma tenemos que

3. $\exists z\ (b = z \times z \wedge \text{Com}(z))$

Sea d tal que

4. $b = d \times d \wedge \text{Com}(d)$

Ya que vale $\text{Com}(c)$, el segundo axioma nos dice que

5. $\forall x\ (x \times c = c \times x)$

Ya que $a = c \times c$ y $b = d \times d$, tenemos que

6. $a \times b = (c \times c) \times (d \times d)$

Pero por el primer axioma (asociatividad) tenemos que

7. $(c \times c) \times (d \times d) = c \times (c \times (d \times d))$

Pero por 5. tenemos que

8. $c \times (c \times (d \times d)) = c \times ((d \times d) \times c)$

Por asociatividad

9. $c \times ((d \times d) \times c) = (c \times (d \times d)) \times c$

Por 5. tenemos que

10. $(c \times (d \times d)) \times c = ((d \times d) \times c) \times c$

Por asociatividad tenemos que

11. $((d \times d) \times c) \times c = (d \times d) \times (c \times c)$

Ya que $a = c \times c$ y $b = d \times d$, tenemos que

12. $(d \times d) \times (c \times c) = b \times a.$

Siguiendo la cadena de igualdades desde 6. hasta 12. tenemos que

13. $a \times b = b \times a$.

Ya que a y b eran elementos arbitrarios, hemos probado que $\forall x \forall y\ x \times y = y \times x$

6.6 Programa

Ahora que hemos generalizado los conceptos de estructura, formula elemental y prueba elemental via el concepto de tipo, podemos enunciar en forma mucho mas general el programa de logica para reticulados cuaterna dado al principio de la seccion.

Programa de logica matematica

(1) Dar un modelo matematico del concepto de formula elemental de tipo τ

(2) Dar una definicion matematica de cuando una formula elemental de tipo τ es verdadera en una estructura de tipo τ para una asignacion dada de valores a las variables libres y a los nombres de constantes fijas de la formula

(3) (Plato gordo) Dar un modelo matematico del concepto de prueba elemental en una teoria elemental de tipo τ. A estos objetos matematicos los llamaremos pruebas formales de tipo τ

(4) (Sublime) Intentar probar matematicamente que nuestro concepto de prueba formal de tipo τ es una correcta modelizacion matematica de la idea intuitiva de prueba elemental en una teoria elemental de tipo τ

Como veremos, los cuatro puntos anteriores pueden ser hechos satisfactoriamente y constituyen el comienzo de la logica matematica con cuantificadores. Cabe aclarar que la realizacion del cuarto punto es realmente sorprendente ya que es un caso de una prueba matematica rigurosa de un hecho que involucra un concepto intuitivo como lo es el de prueba elemental.

El punto (1) se resuelve en la seccion siguiente y si bien produce interesantes conceptos y resultados matematicos su resolucion es rutinaria. El punto (2) es resuelto por Tarski. El punto (3) por Fregue. El (4) es una concecuencia de dos importantes teoremas, el Teorema de Correccion y el Teorema de Completitud de Godel.

6.7 Modelo matematico de la sintaxis elemental

En esta seccion daremos un modelo matematico del concepto de formula elemental de tipo τ. Esto coresponde al punto (1) del programa de logica enunciado anteriormente.

6.7.1 Variables

Las variables usadas en las formulas elementales no estaban del todo especificadas. Para hacer bien preciso este concepto definiremos un conjunto concreto de variables. Sea Var el siguiente conjunto de palabras del alfabeto $\{\mathsf{X}, \mathit{0}, \mathit{1}, ..., \mathit{9}, \mathbf{0}, \mathbf{1}, ..., \mathbf{9}\}$:

$$Var = \{\mathsf{X}\mathbf{1}, \mathsf{X}\mathbf{2}, ..., \mathsf{X}\mathbf{9}, \mathsf{X}\mathit{1}\mathbf{0}, \mathsf{X}\mathit{1}\mathbf{1}, ..., \mathsf{X}\mathit{1}\mathbf{9}, \mathsf{X}\mathit{2}\mathbf{0}, \mathsf{X}\mathit{2}\mathbf{1}, ...\}$$

Es decir el elemento n-esimo de Var es la palabra de la forma $\mathsf{X}\alpha$ donde α es el resultado de reemplazar en la palabra que denota n en notacion decimal, el ultimo numeral por su correspondiente numeral bold y los otros por sus correspondientes italicos. A los elementos de Var los llamaremos *variables.* La razon por la cual usamos numerales italicos y bold es que a los numerales normales los usamos habitualmente en los tipos y sera conveniente que entonces no ocurran en las variables. Ademas tomamos el ultimo simbolo de cada variable en bold para que de esta manera nunca una variable sea una subpalabra de otra variable distinta a ella, lo cual contribuye a simplificar los resultados.

Denotaremos con x_i al i-esimo elemento de Var, para cada $i \in \mathbf{N}$.

6.7.2 Terminos

Dado un tipo τ, definamos recursivamente los conjuntos de palabras T_k^τ, con $k \geq 0$, de la siguiente manera:

$$\begin{array}{rcl} T_0^\tau & = & Var \cup \mathcal{C} \\ T_{k+1}^\tau & = & T_k^\tau \cup \{f(t_1, ..., t_n) : f \in \mathcal{F}_n,\, n \geq 1 \text{ y } t_1, ..., t_n \in T_k^\tau\}. \end{array}$$

Sea

$$T^\tau = \bigcup_{k \geq 0} T_k^\tau$$

Los elementos de T^τ seran llamados *terminos de tipo* τ. Un termino t es llamado *cerrado* si x_i no es subpalabra de t, para cada $i \in \mathbf{N}$. Definamos

$$T_c^\tau = \{t \in T^\tau : t \text{ es cerrado}\}$$

Algunos ejemplos:

(E1) Sea $\tau = (\{\text{uno}, \text{doli}\}, \{\text{MAS}, \text{P}\}, \{\text{Her}\}, a)$, con a dado por $a(\text{MAS}) = 4$, $a(\text{P}) = 1$ y $a(\text{Her}) = 3$. Entonces

(a) Las palabras uno, doli y $\mathsf{X}\mathit{15666}\mathbf{9}$ son terminos de tipo τ ya que pertenecen a T_0^τ

(b) $\text{MAS}(\text{uno}, \text{doli}, \mathsf{X}\mathit{1}\mathbf{9}, \mathsf{X}\mathbf{5})$ y $\text{P}(\text{uno})$ son terminos de tipo τ ya que pertenecen a T_1^τ (por que?)

(c) Las palabras

$$\text{P(P(uno))} \qquad \text{MAS}(\text{P}(\mathsf{X}\mathbf{4}), \text{doli}, \mathsf{X}\mathit{1}\mathbf{9}, \mathsf{X}\mathbf{5})$$

son terminos de tipo τ ya que pertenecen a T_2^τ

(d) $\text{P}(\text{MAS}(\text{P}(\mathsf{X}\mathbf{4}), \text{MAS}(\mathsf{X}\mathbf{1}, \mathsf{X}\mathbf{2}, \mathsf{X}\mathbf{3}, \mathsf{X}\mathbf{4}), \mathsf{X}\mathit{1}\mathbf{9}, \mathsf{X}\mathbf{5}))$ es un termino ya que pertenece a T_3^τ

(e) uno, doli, P(uno) y $\text{MAS}(\text{uno}, \text{doli}, \text{doli}, \text{doli})$ son terminos cerrados de tipo τ

Lo que debe quedar claro es que como objetos matematicos los terminos son meras palabras, por ejemplo $\text{MAS}(\text{uno}, \text{doli}, \mathsf{X}\mathit{1}\mathbf{9}, \mathsf{X}\mathbf{5})$ es una palabra (de longitud 20)

(E2) Sea $\tau = (\{0,1\}, \{+, \times, \uparrow\}, \emptyset, a)$, con a dado por $a(+) = 2$, $a(\times) = 3$ y $a(\uparrow) = 1$. Entonces

$$\mathsf{X}\mathit{111}\mathbf{9} \qquad 0 \qquad 1 \qquad +(+(\uparrow(\mathsf{X}\mathbf{4}), \times(\mathsf{X}\mathbf{2}, 1, 0)), \times(1, \mathsf{X}\mathbf{2}, \mathsf{X}\mathbf{3}))$$

son terminos de tipo τ. Tambien $\uparrow(+(\uparrow(0), \times(0,1,0)))$ es un termino cerrado de tipo τ

(E3) Sea $\tau = (\emptyset, \{\mathsf{s}, \mathsf{i}\}, \emptyset, \{(\mathsf{s}, 2), (\mathsf{i}, 2)\})$ el tipo de los reticulados terna. Entonces

$$\mathsf{s}(\mathsf{X}\mathbf{2}, \mathsf{X}\mathbf{3}) \qquad \mathsf{s}(\mathsf{s}(\mathsf{X}\mathbf{4}, \mathsf{X}\mathit{1}\mathbf{4}), \mathsf{i}(\mathsf{X}\mathbf{2}, \mathsf{X}\mathit{111}\mathbf{9}))$$

son terminos de tipo τ. No hay terminos cerrados de tipo τ. Cabe destacar que $\mathsf{X}\mathbf{2}\ \mathsf{s}\ \mathsf{X}\mathbf{3}$ no es un termino de tipo τ aunque, como veremos en los ejercicios esto no es trivial de la definicion de termino y requiere de una demostracion.

El siguiente lema es la herramienta basica para probar propiedades de los terminos.

Lemma 175 *Supongamos $t \in T_k^\tau$, con $k \geq 1$. Entonces ya sea $t \in Var \cup \mathcal{C}$ o $t = f(t_1, ..., t_n)$, con $f \in \mathcal{F}_n$, $n \geq 1$ y $t_1, ..., t_n \in T_{k-1}^\tau$.*

Proof. Por induccion en k.

CASO $k = 1$: Es directo ya que por definicion

$$T_1^\tau = Var \cup \mathcal{C} \cup \{f(t_1, ..., t_n) : f \in \mathcal{F}_n,\ n \geq 1 \text{ y } t_1, ..., t_n \in T_0^\tau\}.$$

CASO $k \Rightarrow k+1$: Sea $t \in T_{k+1}^\tau$. Por definicion de T_{k+1}^τ tenemos que $t \in T_k^\tau$ o $t = f(t_1, ..., t_n)$ con $f \in \mathcal{F}_n$, $n \geq 1$ y $t_1, ..., t_n \in T_k^\tau$. Si se da que $t \in T_k^\tau$, entonces podemos aplicar hipotesis inductiva y usar que $T_{k-1}^\tau \subseteq T_k^\tau$. Esto completa el caso. ■

Algunos ejemplos de propiedades de los terminos las cuales se pueden probar facilmente usando el lema anterior son

- Si $t \in T^\tau$ es tal que en t ocurre el simbolo), entonces $t = f(t_1, ..., t_n)$ con $f \in \mathcal{F}_n$, $n \geq 1$ y $t_1, ..., t_n \in T^\tau$.

- Ningun termino comienza con un simbolo del alfabeto $\{\mathit{0}, \mathit{1}, ..., \mathit{9}\}$

- Si $t \in T^\tau$ comienza con X entonces $t \in Var$

- Si $t \in T^\tau$ y $[t]_i =$), con $i < |t|$, entonces $[t]_{i+1} =$, o $[t]_{i+1} =$)

- Si $t \in T^\tau$, entonces $|t|_(= |t|_)$.

Una posible forma de probar que una palabra dada no es un termino es encontrar una propiedad que posean todos los terminos la cual no cumpla dicha palabra. Por ejemplo si $\tau = (\emptyset, \{glp\}, \emptyset, a)$, con $a(glp) = 1$, la palabra $\alpha = glp((\mathsf{X}\mathit{13}\mathbf{3})$ no es un termino ya que $|\alpha|_(\neq |\alpha|_)$.

Unicidad de la lectura de terminos Definamos conjuntos Bal_k, con $k \geq 1$ de la siguiente manera:

$$\begin{array}{rcl} Bal_1 & = & \{()\} \\ Bal_{k+1} & = & Bal_k \cup \{(b_1 ... b_n) : b_1, ..., b_n \in Bal_k, n \geq 1\}. \end{array}$$

Sea

$$Bal = \bigcup_{k \geq 1} Bal_k$$

Recordemos que β es un *tramo inicial (propio)* de α si hay una palabra γ tal que $\alpha = \beta\gamma$ (y $\beta \notin \{\varepsilon, \alpha\}$). En forma similar se define *tramo final (propio).*

Lemma 176 *Sea $b \in Bal$. Se tiene:*

(1) $|b|_(- |b|_) = 0$

(2) Si x es tramo inicial propio de b, entonces $|x|_(- |x|_) > 0$

(3) Si x es tramo final propio de b, entonces $|x|_(- |x|_) < 0$

Proof. Probaremos por induccion en k, que valen (1), (2) y (3) para cada $b \in Bal_k$. El caso $k = 1$ es trivial. Supongamos $b \in Bal_{k+1}$. Si $b \in Bal_k$, se aplica directamente HI. Supongamos entonces que $b = (b_1 ... b_n)$, con $b_1, ..., b_n \in Bal_k$, $n \geq 1$. Por HI, $b_1, ..., b_n$ cumplen (1) por lo cual b cumple (1). Veamos que b cumple (2). Sea x un tramo inicial propio de b. Notese que x es de la forma $x = (b_1 ... b_i x_1$ con $0 \leq i \leq n - 1$ y x_1 un tramo inicial de b_{i+1} (en el caso $i = 0$ interpretamos $b_1 ... b_i = \varepsilon$). Pero entonces ya que

$$|x|_(- |x|_) = 1 + \left(\sum_{j=1}^{i} |b_j|_(- |b_j|_) \right) + |x_1|_(- |x_1|_)$$

tenemos que por HI, se da que $|x|_(- |x|_) > 0$. En forma analoga se puede ver que b cumple (3). ■

Dado un alfabeto Σ tal que (y) pertenecen a Σ, definamos $del : \Sigma^* \rightarrow \Sigma^*$, de la siguiente manera

$$\begin{array}{rcl} del(\varepsilon) & = & \varepsilon \\ del(\alpha a) & = & del(\alpha)a, \text{ si } a \in \{(,)\} \\ del(\alpha a) & = & del(\alpha), \text{ si } a \in \Sigma - \{(,)\} \end{array}$$

Lemma 177 *$del(xy) = del(x)del(y)$, para todo $x, y \in \Sigma^*$*

Lemma 178 *Supongamos que Σ es tal que $T^\tau \subseteq \Sigma^*$. Entonces $del(t) \in Bal$, para cada $t \in T^\tau - (Var \cup \mathcal{C})$*

Notese que en la definicion de tipo se exige que nunca un nombre de cte sea subpalabra de otro nombre de cte, lo cual garantiza que nunca puede ser un nombre de cte un tramo inicial o final propio de otro nombre de cte. Lo que si puede suceder es que un tramo final propio de un nombre de cte c sea un tramo inicial propio de otro nombre de cte d. Mas formalmente puede suceder que haya palabras x, y, z, las tres distintas de ε tales que $c = xy$ y $d = yz$. En tal caso solemos decir que las palabras c y d se *mordizquean.* Por ejemplo si $\tau = (\{\text{uno,noli}\}, \emptyset, \emptyset, \emptyset)$, es facil ver que τ es un tipo y que uno y noli se mordizquean. El lema siguiente nos dice que este es el unico caso de mordizqueo de terminos.

Lemma 179 *(Mordizqueo de Terminos) Sean $s, t \in T^\tau$ y supongamos que hay palabras x, y, z, con $y \neq \varepsilon$ tales que $s = xy$ y $t = yz$. Entonces $x = z = \varepsilon$ o $s, t \in \mathcal{C}$. En particular si un termino es tramo inicial o final de otro termino, entonces dichos terminos son iguales.*

Proof. Supongamos $s \in \mathcal{C}$. Ya que $y \neq \varepsilon$ tenemos que t debe comenzar con un simbolo que ocurre en un nombre de cte, lo cual dice que t no puede ser ni una variable ni de la forma $g(t_1, ..., t_m)$, es decir $t \in \mathcal{C}$. Supongamos $s \in Var$. Si $x \neq \varepsilon$ tenemos que t debe comenzar con alguno de los siguientes simbolos

0 1 ... 9 **0 1** ... **9**

lo cual es absurdo. O sea que $x = \varepsilon$ y por lo tanto t debe comenzar con X. Pero esto dice que $t \in Var$ de lo que sigue facilmente que $z = \varepsilon$. Supongamos entonces que s es de la forma $f(s_1, ..., s_n)$. Ya que) debe ocurrir en t, tenemos que t es de la forma $g(t_1, ..., t_m)$. O sea que $del(s), del(t) \in Bal$. Ya que) ocurre en y, $del(y) \neq \varepsilon$. Tenemos tambien que

$$\begin{array}{rcl} del(s) & = & del(x)del(y) \\ del(t) & = & del(y)del(z) \end{array}$$

La primera igualdad, por (1) y (3) del Lema 176, nos dice que

$$|del(y)|_{(} - |del(y)|_{)} \leq 0,$$

y la segunda que

$$|del(y)|_{(} - |del(y)|_{)} \geq 0,$$

por lo cual

$$|del(y)|_{(} - |del(y)|_{)} = 0$$

Pero entonces (3) del Lema 176 nos dice que $del(y)$ no puede ser tramo final propio de $del(s)$, por lo cual debe suceder que $del(y) = del(s)$, ya que $del(y) \neq \varepsilon$. Claramente entonces obtenemos que $del(x) = \varepsilon$. Similarmente se puede ver que $del(z) = \varepsilon$. Ya que que t termina con) tenemos que $z = \varepsilon$. O sea que $f(s_1, ..., s_n) = xg(t_1, ..., t_m)$ con $del(x) = \varepsilon$, de lo que se saca que $f = xg$ ya que (no ocurre en x. De la definicion de tipo se desprende que $x = \varepsilon$. ■

Theorem 180 *(Lectura unica de terminos). Dado $t \in T^\tau$ se da una de las siguientes:*

(1) $t \in Var \cup \mathcal{C}$

(2) Hay unicos $n \geq 1,\ f \in \mathcal{F}_n,\ t_1, ..., t_n \in T^\tau$ *tales que* $t = f(t_1, ..., t_n)$.

Proof. En virtud del Lema 175 solo nos falta probar la unicidad en el punto (2). Supongamos que

$$t = f(t_1, ..., t_n) = g(s_1, ..., s_m)$$

con $n, m \geq 1,\ f \in \mathcal{F}_n,\ g \in \mathcal{F}_m,\ t_1, ..., t_n, s_1, ..., s_m \in T^\tau$. Notese que $f = g$. O sea que $n = m = a(f)$. Notese que t_1 es tramo inicial de s_1 o s_1 es tramo inicial de t_1, lo cual por el lema anterior nos dice que $t_1 = s_1$. Con el mismo razonamiento podemos probar que debera suceder $t_2 = s_2, ..., t_n = s_n$. ■

El teorema anterior es importante ya que nos permite definir recursivamente funciones con dominio contenido en T^τ. Por ejemplo podemos definir una funcion $F : T^\tau \to T^\tau$, de la siguiente manera:

- $F(c) = c$, para cada $c \in \mathcal{C}$
- $F(v) = v$, para cada $v \in Var$
- $F(f(t_1, ..., t_n)) = f(F(t_1), ..., F(t_n))$, si $f \in \mathcal{F}_n$, con $n \neq 2$
- $F(f(t_1, t_2)) = f(t_2, t_1)$, si $f \in \mathcal{F}_2$.

Notese que si la unicidad de la lectura no fuera cierta, entonces las ecuaciones anteriores no estarian definiendo en forma correcta una funcion ya que el valor de la imagen de un termino t estaria dependiendo de cual descomposicion tomemos para t.

Subterminos Sean $s, t \in T^\tau$. Diremos que s es *subtermino* (*propio*) de t si (no es igual a t y) s es subpalabra de t. A continuacion veremos de que manera ocurren los subterminos de un termino. Para esto recordemos un poco el concepto de ocurrencia.

Dadas palabras $\alpha, \beta \in \Sigma^*$, con $|\alpha|, |\beta| \geq 1$, y un natural $i \in \{1, ..., |\beta|\}$, se dice que α *ocurre a partir de i en β* cuando se de que existan palabras δ, γ tales que $\beta = \delta\alpha\gamma$ y $|\delta| = i - 1$. Intuitivamente hablando α ocurre a partir de i en β cuando se de que si comensamos a leer desde el lugar i-esimo de β en adelante, leeremos la palabra α completa y luego posiblemente seguiran otros simbolos.

Notese que una palabra α puede ocurrir en β, a partir de i, y tambien a partir de j, con $i \neq j$. En virtud de esto, hablaremos de las distintas ocurrencias de α en β. Por ejemplo hay dos ocurrencias de la palabra *aba* en la palabra

$$ccccccabaccccabaccccc$$

y tambien hay dos ocurrencias de la palabra *aba* en la palabra

$$ccccccababaccccccccc$$

En el primer caso diremos que dichas ocurrencias de *aba* son *disjuntas*, en cambio en el segundo caso puede apreciarse que las dos ocurrencias se superponen en una posicion. No definiremos en forma matematica precisa el concepto de ocurrencia pero lo describiremos con ejemplos de manera que el lector no tendra problemas en comprenderlo y manejarlo en forma correcta.

A veces diremos que una ocurrencia esta *contenida* o *sucede* dentro de otra. Por ejemplo la segunda ocurrencia de *ab* en $babbbfabcccfabccc$ esta contenida en la primer ocurrencia de $fabc$ en $babbbfabcccfabccc$. Tambien haremos *reemplazos* de ocurrencias por palabras. Por ejemplo el resultado de reemplazar la primer ocurrencia de *abb* en $ccabbgfgabbgg$ por *oolala* es la palabra $ccoolalagfgabbgg$. El resultado de reemplazar todas las ocurrencias de *aba* en $ccabagfgabaggaba$ por $$ es la palabra $cc\$\$gfg\$\$gg\$\$$. En algunos casos deberemos especificar que los reemplazos se hagan *simultaneamente*. Por ejemplo hablaremos del resultado de reemplazar en γ, simultaneamente, todas las ocurrencias de α_1 por β_1 y todas las de α_2 por β_2. Aqui la aclaracion de simultaneidad es importante ya que si primero reemplazaramos las ocurrencias de α_1 por β_1 y despues las de α_2 por β_2, el resultado puede cambiar porque en β_1 puede haber ocurrencias de α_2. Dejamos al lector dar un ejemplo en el cual el reemplazo secuencial y el simultaneo dan distintos resultados.

Lemma 181 *Sean $r, s, t \in T^\tau$.*

(a) Si $s \neq t = f(t_1, ..., t_n)$ y s ocurre en t, entonces dicha ocurrencia sucede dentro de algun t_j, $j = 1, ..., n$.

(b) Si r, s ocurren en t, entonces dichas ocurrencias son disjuntas o una ocurre dentro de otra. En particular, las distintas ocurrencias de r en t son disjuntas.

(c) *Si t' es el resultado de reemplazar una ocurrencia de s en t por r, entonces $t' \in T^{\tau}$.*

Proof. (a) Supongamos la ocurrencia de s comienza en algun t_j. Entonces el Lema 179 nos conduce a que dicha ocurrencia debera estar contenida en t_j. Veamos que la ocurrencia de s no puede ser a partir de un $i \in \{1, ..., |f|\}$. Supongamos lo contrario. Tenemos entonces que s debe ser de la forma $g(s_1, ..., s_m)$ ya que no puede estar en $Var \cup \mathcal{C}$. Notese que $i \neq 1$ ya que en caso contrario s seria un tramo inicial propio de t. Pero entonces g debe ser un tramo final propio de f, lo cual es absurdo. Ya que s no puede comenzar con parentesis o coma, hemos contemplado todos los posibles casos de comienzo de la ocurrencia de s en t.

(b) y (c) pueden probarse por induccion, usando (a). ■

Nota: Es importante notar que si bien no hemos definido en forma presisa el concepto de ocurrencia o de reemplazo de ocurrencias, la prueba del lema anterior es rigurosa en el sentido de que solo usa propiedades del concepto de ocurrencia y reemplazo de ocurrencias las cuales deberan ser comunes a cualquier definicion o formulacion matematica que se hiciera de aquellos conceptos. En este caso, es posible dar una defincion presisa y satisfactoria de dichos conceptos aunque para otros conceptos tales como el de prueba absoluta de consistencia, aun no se ha encontrado una formulacion matematica adecuada.

6.7.3 Formulas

Sea τ un tipo. Las palabras de alguna de las siguientes dos formas

$$(t \equiv s), \text{ con } t, s \in T^{\tau}$$
$$r(t_1, ..., t_n), \text{ con } r \in \mathcal{R}_n,\ n \geq 1 \text{ y } t_1, ..., t_n \in T^{\tau}$$

seran llamadas *formulas atomicas de tipo* τ. Por ejemplo si $\tau = (\{\text{uno}, \text{doli}\}, \{\text{MAS}, \text{P}\}, \{\text{Her}\}, a)$, con a dado por $a(\text{MAS}) = 4$, $a(\text{P}) = 1$ y $a(\text{Her}) = 3$, entonces

- $(\text{uno} \equiv \text{doli})$
- $(\mathsf{X}\mathit{15666}\mathbf{9} \equiv \text{doli})$
- $\text{Her}(\text{uno}, \mathsf{X}\mathbf{4}, \text{doli})$
- $(\text{MAS}(\text{uno}, \text{doli}, \mathsf{X}\mathit{1}\mathbf{9}, \mathsf{X}\mathbf{5}) \equiv \text{uno})$
- $\text{Her}(\text{P}(\text{P}(\text{uno})), \text{MAS}(\text{P}(\mathsf{X}\mathbf{4}), \text{doli}, \mathsf{X}\mathit{1}\mathbf{9}, \mathsf{X}\mathbf{5}), \mathsf{X}\mathit{1}\mathbf{9})$

son formulas atomicas de tipo τ.

Dado un tipo τ, definamos recursivamente los conjuntos de palabras F_k^τ, con $k \geq 0$, de la siguiente manera:

$$\begin{array}{rcl} F_0^\tau & = & \{\text{formulas atomicas de tipo } \tau\} \\ F_{k+1}^\tau & = & F_k^\tau \cup \{\neg\varphi : \varphi \in F_k^\tau\} \cup \{(\varphi \vee \psi) : \varphi, \psi \in F_k^\tau\} \cup \\ & & \qquad \{(\varphi \wedge \psi) : \varphi, \psi \in F_k^\tau\} \cup \{(\varphi \rightarrow \psi) : \varphi, \psi \in F_k^\tau\} \cup \\ & & \qquad\qquad \{(\varphi \leftrightarrow \psi) : \varphi, \psi \in F_k^\tau\} \cup \{\forall v\varphi : \varphi \in F_k^\tau \text{ y } v \in Var\} \cup \\ & & \qquad\qquad\qquad \{\exists v\varphi : \varphi \in F_k^\tau \text{ y } v \in Var\} \end{array}$$

Sea

$$F^\tau = \bigcup_{k \geq 0} F_k^\tau$$

Los elementos de F^τ seran llamados *formulas de tipo* τ.

Algunos ejemplos:

(E1) Sea $\tau = (\{\text{uno}, \text{doli}\}, \{\text{MAS}, \text{P}\}, \{\text{Her}\}, a)$, con a dado por $a(\text{MAS}) = 4$, $a(\text{P}) = 1$ y $a(\text{Her}) = 3$. Entonces

   (a) $\neg((\mathsf{X}\mathbf{1} \equiv \mathsf{X}\mathbf{2}) \wedge \text{Her}(\text{P}(\text{doli}), \text{doli}, \mathsf{X}\mathit{1}\mathbf{9}))$
   (b) $\exists\mathsf{X}\mathbf{9}\text{Her}(\text{doli}, \text{doli}, \mathsf{X}\mathbf{9})$
   (c) $\exists\mathsf{X}\mathbf{9}\neg(\text{uno} \equiv \text{doli})$
   (d) $\neg\exists\mathsf{X}\mathbf{9}\forall\mathsf{X}\mathbf{7}(\text{Her}(\mathsf{X}\mathbf{9}, \text{doli}, \mathsf{X}\mathbf{7}) \rightarrow (\text{P}(\text{doli}) \equiv \mathsf{X}\mathbf{7}))$
   (e) $\forall\mathsf{X}\mathit{555}\mathbf{9}\forall\mathsf{X}\mathbf{7}\exists\mathsf{X}\mathit{5}\mathbf{1}(\text{MAS}(\text{uno}, \text{doli}, \mathsf{X}\mathit{1}\mathbf{9}, \mathsf{X}\mathbf{5}) \equiv \text{uno}) \rightarrow \text{Her}(\text{doli}, \text{doli}, \text{doli}))$

   son formulas de tipo τ

(E2) Sea $\tau = (\{0,1\}, \{\mathsf{s}, \mathsf{i}\}, \{\leq\}, \{(\mathsf{s}, 2), (\mathsf{i}, 2), (\leq, 2)\})$ el tipo de los reticulados cuaterna. Entonces

   (a) $\leq(1,0)$
   (b) $\leq(\mathsf{X}\mathbf{1}, \mathsf{X}\mathbf{2})$
   (c) $\neg(\mathsf{s}(\mathsf{X}\mathbf{2}, \mathsf{X}\mathbf{1}) \equiv \mathsf{X}\mathbf{2}))$
   (d) $\forall\mathsf{X}\mathbf{2}\forall\mathsf{X}\mathbf{1}\leq(\mathsf{X}\mathbf{2}, \mathsf{s}(\mathsf{X}\mathbf{2}, \mathsf{X}\mathbf{1}))$
   (e) $((\mathsf{i}(\mathsf{X}\mathbf{1}, \mathsf{X}\mathbf{2}) \equiv 0) \wedge (\mathsf{s}(\mathsf{X}\mathbf{1}, \mathsf{X}\mathbf{2}) \equiv 1))$
   (f) $\forall\mathsf{X}\mathbf{9}\exists\mathsf{X}\mathbf{1}((0 \equiv \mathsf{X}\mathbf{1}) \rightarrow \exists\mathsf{X}\mathbf{1}\neg\leq(\mathsf{X}\mathbf{2}, \mathsf{s}(\mathsf{X}\mathbf{2}, \mathsf{X}\mathbf{1})))$

   son formulas de tipo τ. Cabe destacar que $(\mathsf{X}\mathbf{1} \leq \mathsf{X}\mathbf{2})$ no es una formula de tipo τ aunque, como veremos en los ejercicios esto no es trivial de la definicion de formula y requiere de una demostracion.

El siguiente lema es la herramienta basica que usaremos para probar propiedades acerca de los elementos de F^τ.

Lemma 182 *Supongamos $\varphi \in F_k^\tau$, con $k \geq 1$. Entonces φ es de alguna de las siguientes formas*

$\varphi = (t \equiv s)$, *con* $t, s \in T^\tau$.
$\varphi = r(t_1, ..., t_n)$, *con* $r \in \mathcal{R}_n$, $t_1, ..., t_n \in T^\tau$
$\varphi = (\varphi_1 \eta \varphi_2)$, *con* $\eta \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$, $\varphi_1, \varphi_2 \in F_{k-1}^\tau$
$\varphi = \neg\varphi_1$, *con* $\varphi_1 \in F_{k-1}^\tau$
$\varphi = Qv\varphi_1$, *con* $Q \in \{\forall, \exists\}$, $v \in Var$ *y* $\varphi_1 \in F_{k-1}^\tau$.

Proof. Induccion en k. ∎

Unicidad de la lectura de formulas Tal como para el caso de terminos veremos que las formulas tambien tienen su unicidad de lectura.

Lemma 183 *Sea τ un tipo.*

(a) Supongamos que Σ es tal que $F^\tau \subseteq \Sigma^*$. Entonces $del(\varphi) \in Bal$, para cada $\varphi \in F^\tau$.

(b) Sea $\varphi \in F_k^\tau$, con $k \geq 0$. Existen $x \in (\{\neg\} \cup \{Qv : Q \in \{\forall, \exists\}$ y $v \in Var\})^*$ y $\varphi_1 \in F^\tau$ tales que $\varphi = x\varphi_1$ y φ_1 es de la forma $(\psi_1 \eta \psi_2)$ o atomica. En particular toda formula termina con el simbolo).

Proof. (b) Induccion en k. El caso $k = 0$ es trivial. Supongamos (b) vale para cada $\varphi \in F_k^\tau$ y sea $\varphi \in F_{k+1}^\tau$. Hay varios casos de los cuales haremos solo dos

CASO $\varphi = (\psi_1 \eta \psi_2)$, con $\psi_1, \psi_2 \in F_k^\tau$ y $\eta \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$.
Podemos tomar $x = \varepsilon$ y $\varphi_1 = \varphi$.

CASO $\varphi = Qx_i\psi$, con $\psi \in F_k^\tau$, $i \geq 1$ y $Q \in \{\forall, \exists\}$.
Por HI hay $\bar{x} \in (\{\neg\} \cup \{Qv : Q \in \{\forall, \exists\}$ y $v \in Var\})^*$ y $\psi_1 \in F^\tau$ tales que $\psi = \bar{x}\psi_1$ y ψ_1 es de la forma $(\gamma_1 \eta \gamma_2)$ o atomica. Entonces es claro que $x = Qx_i\bar{x}$ y $\varphi_1 = \psi_1$ cumplen (b). ∎

Lemma 184 *Ninguna formula es tramo final propio de una formula atomica, es decir, si $\varphi = x\psi$, con $\varphi \in F_0^\tau$ y ,$\psi \in F^\tau$, entonces $x = \varepsilon$.*

Proof. Si φ es de la forma $(t \equiv s)$, entonces $|del(y)|_(- |del(y)|_) < 0$ para cada tramo final propio y de φ, lo cual termina el caso ya que $del(\psi)$ es balanceada. Supongamos entonces $\varphi = r(t_1, ..., t_n)$. Notese que ψ no puede ser tramo final de $t_1, ..., t_n)$ ya que $del(\psi)$ es balanceada y $|del(y)|_(- |del(y)|_) < 0$ para cada tramo final y de $t_1, ..., t_n)$. Es decir que $\psi = y(t_1, ..., t_n)$, para algun tramo final y de r. Ya que en ψ no ocurren cuantificadores ni nexos ni el simbolo $\equiv$ el Lema 182 nos dice $\psi = \tilde{r}(s_1, ..., s_m)$, con $\tilde{r} \in \mathcal{R}_m$, $m \geq 1$ y $s_1, ..., s_m \in T^\tau$. Ahora es facil usando un argumento paresido al usado en la prueba del Teorema 180 concluir que $m = n$, $s_i = t_i$, $i = 1, ..., n$ y $\tilde{r}$ es tramo final de r. Por (3) de la definicion de tipo tenemos que $\tilde{r} = r$ lo cual nos dice que $\varphi = \psi$ y $x = \varepsilon$ ∎

Lemma 185 *Si* $\varphi = x\psi$*, con* $\varphi, \psi \in F^\tau$ *y* x *sin parentesis, entonces* $x \in (\{\neg\} \cup \{Qv : Q \in \{\forall, \exists\}$ *y* $v \in Var\})^*$

Proof. Por induccion en el k tal que $\varphi \in F_k^\tau$. El caso $k = 0$ es probado en el lema anterior. Asumamos que el resultado vale cuando $\varphi \in F_k^\tau$ y veamos que vale cuando $\varphi \in F_{k+1}^\tau$. Mas aun supongamos $\varphi \in F_{k+1}^\tau - F_k^\tau$. Primero haremos el caso en que $\varphi = Qv\varphi_1$, con $Q \in \{\forall, \exists\}$, $v \in Var$ y $\varphi_1 \in F_k^\tau$. Supongamos $x \neq \varepsilon$. Ya que ψ no comienza con simbolos de v, tenemos que ψ debe ser tramo final de φ_1 lo cual nos dice que hay una palabra x_1 tal que $x = Qvx_1$ y $\varphi_1 = x_1\psi$. Por HI tenemos que $x_1 \in (\{\neg\} \cup \{Qv : Q \in \{\forall, \exists\}$ y $v \in Var\})^*$ con lo cual $x \in (\{\neg\} \cup \{Qv : Q \in \{\forall, \exists\}$ y $v \in Var\})^*$. El caso en el que $\varphi = \neg\varphi_1$ con $\varphi_1 \in F_k^\tau$, es similar. Note que no hay mas casos posibles ya que φ no puede comenzar con (porque en x no ocurren parentesis por hipotesis ■

Proposition 186 *Si* $\varphi, \psi \in F^\tau$ *y* x, y, z *son tales que* $\varphi = xy$, $\psi = yz$ *y* $y \neq \varepsilon$*, entonces* $z = \varepsilon$ *y* $x \in (\{\neg\} \cup \{Qv : Q \in \{\forall, \exists\}$ *y* $v \in Var\})^*$*. En particular ningun tramo inicial propio de una formula es una formula.*

Proof. Ya que φ termina con) tenemos que $del(y) \neq \varepsilon$. Por un lema anterior tenemos que $del(\varphi), del(\psi) \in Bal$. Ademas

$$\begin{array}{rcl} del(\varphi) & = & del(x)del(y) \\ del(\psi) & = & del(y)del(z) \end{array}$$

La primera igualdad, por (1) y (3) del Lema 176, nos dice que

$$|del(y)|_(- |del(y)|_) \leq 0,$$

y la segunda que

$$|del(y)|_(- |del(y)|_) \geq 0,$$

por lo cual

$$|del(y)|_(- |del(y)|_) = 0$$

Pero entonces (3) del Lema 176 nos dice que $del(y)$ no puede ser tramo final propio de $del(\varphi)$, por lo cual debe suceder que $del(y) = del(\varphi)$, ya que $del(y) \neq \varepsilon$. Claramente entonces obtenemos que $del(x) = \varepsilon$. Similarmente se puede ver que $del(z) = \varepsilon$. Pero ψ termina con) lo cual nos dice que $z = \varepsilon$. Es decir que $\varphi = x\psi$. Por el lema anterior tenemos que $x \in (\{\neg\} \cup \{Qv : Q \in \{\forall, \exists\}$ y $v \in Var\})^*$ ■

Theorem 187 *(Lectura unica de formulas) Dada* $\varphi \in F^\tau$ *se da una y solo una de las siguientes:*

(1) $\varphi = (t \equiv s)$*, con* $t, s \in T^\tau$

(2) $\varphi = r(t_1, ..., t_n)$*, con* $r \in \mathcal{R}_n$*,* $t_1, ..., t_n \in T^\tau$

(3) $\varphi = (\varphi_1 \eta \varphi_2)$, *con* $\eta \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$, $\varphi_1, \varphi_2 \in F^\tau$

(4) $\varphi = \neg\varphi_1$, *con* $\varphi_1 \in F^\tau$

(5) $\varphi = Qv\varphi_1$, *con* $Q \in \{\forall, \exists\}$, $\varphi_1 \in F^\tau$ *y* $v \in Var$.

Mas aun, en los puntos (1), (2), (3), (4) y (5) tales descomposiciones son unicas.

Proof. Si una formula φ satisface (1), entonces φ no puede contener simbolos del alfabeto $\{\wedge, \vee, \rightarrow, \leftrightarrow\}$ lo cual garantiza que φ no puede satisfacer (3). Ademas φ no puede satisfacer (2) o (4) o (5) ya que φ comienza con (. En forma analoga se puede terminar de ver que las propiedades (1),...,(5) son excluyentes.

La unicidad en las descomposiciones de (4) y (5) es obvia. La de (3) se desprende facilmente del lema anterior y la de los puntos (1) y (2) del lema analogo para terminos. ■

Subformulas Una formula φ sera llamada una *subformula* (*propia*) de una formula ψ, cuando φ (sea no igual a ψ y) sea subpalabra de ψ.

Lemma 188 *Sea* τ *un tipo.*

(a) Las formulas atomicas no tienen subformulas propias.

(b) Si φ *ocurre propiamente en* $(\psi\eta\gamma)$, *entonces tal ocurrencia es en* ψ *o en* γ.

(c) Si φ *ocurre propiamente en* $\neg\psi$, *entonces tal ocurrencia es en* ψ.

(d) Si φ *ocurre propiamente en* $Qx_k\psi$, *entonces tal ocurrencia es en* ψ.

(e) Si φ_1, φ_2 *ocurren en* φ, *entonces dichas ocurrencias son disjuntas o una contiene a la otra.*

(f) Si λ' *es el resultado de reemplazar alguna ocurrencia de* φ *en* λ *por* ψ, *entonces* $\lambda' \in F^\tau$.

Proof. Ejercicio. ■

Variables libres Recordemos que dadas palabras $\alpha, \beta \in \Sigma^*$, con $|\alpha|, |\beta| \geq 1$, y un natural $i \in \{1, ..., |\beta|\}$, se dice que α *ocurre a partir de* i *en* β cuando se de que existan palabras δ, γ tales que $\beta = \delta\alpha\gamma$ y $|\delta| = i - 1$. Intuitivamente hablando α ocurre a partir de i en β cuando se de que si comensamos a leer desde el lugar i-esimo de β, en adelante, entonces leeremos la palabra α completa y luego posiblemente seguiran otros simbolos.

Definamos recursivamente la relacion "*v ocurre libremente en φ a partir de i*", donde $v \in Var$, $\varphi \in F^\tau$ y $i \in \{1, ..., |\varphi|\}$, de la siguiente manera:

(1) Si φ es atomica, entonces v ocurre libremente en φ a partir de i sii v ocurre en φ a partir de i

(2) Si $\varphi = (\varphi_1 \eta \varphi_2)$, entonces v ocurre libremente en φ a partir de i sii se da alguna de las siguientes

   (a) v ocurre libremente en φ_1 a partir de $i - 1$

   (b) v ocurre libremente en φ_2 a partir de $i - |(\varphi_1 \eta|$

(3) Si $\varphi = \neg\varphi_1$, entonces v ocurre libremente en φ a partir de i sii v ocurre libremente en φ_1 a partir de $i - 1$

(4) Si $\varphi = Qw\varphi_1$, entonces v ocurre libremente en φ a partir de i sii $v \neq w$ y v ocurre libremente en φ_1 a partir de $i - |Qw|$

Dados $v \in Var$, $\varphi \in F^\tau$ y $i \in \{1, ..., |\varphi|\}$, diremos que "*$v$ ocurre acotadamente en φ a partir de i*" cuando v ocurre en φ a partir de i y v no ocurre libremente en φ a partir de i.

Algunos ejemplos:

1. Sea $\tau = (\{\text{uno}, \text{doli}\}, \{\text{MAS}, \text{P}\}, \{\text{Her}\}, a)$, con a dado por $a(\text{MAS}) = 4$, $a(\text{P}) = 1$ y $a(\text{Her}) = 3$.

   (a) $\mathsf{X}\mathbf{9}$ ocurre libremente en $\text{Her}(\text{doli}, \text{doli}, \mathsf{X}\mathbf{9})$ a partir de 15

   (b) $\mathsf{X}\mathbf{9}$ ocurre acotadamente en $\exists\mathsf{X}\mathbf{9}\text{Her}(\text{doli}, \text{doli}, \mathsf{X}\mathbf{9})$ a partir de 2 y de 18

   (c) $\mathsf{X}\mathbf{2}$ ocurre libremente en $(\exists\mathsf{X}\mathbf{2}\text{Her}(\mathsf{X}\mathbf{2}, \mathsf{X}\mathbf{7}, \text{uno}) \rightarrow \text{Her}(\mathsf{X}\mathbf{2}, \mathsf{X}\mathbf{7}, \text{uno}))$ a partir de 16 y acotadamente a partir de 3 y 7.

   (d) Sea $\varphi = ((\mathsf{X}\mathbf{1} \equiv \mathsf{X}\mathbf{2}) \wedge \exists\mathsf{X}\mathbf{2}\text{Her}(\text{P}(\text{doli}), \text{doli}, \mathsf{X}\mathbf{2}))$. La variable $\mathsf{X}\mathbf{2}$ ocurre libremente en φ a partir de 6 y ocurre acotadamente en φ a partir de 11 y de 30.

Dada una formula φ, sea

$$Li(\varphi) = \{v \in Var : \text{hay un } i \text{ tal que } v \text{ ocurre libremente en } \varphi \text{ a partir de } i\}.$$

Los elementos de $Li(\varphi)$ seran llamados *variables libres de* φ. Por ejemplo, si φ es la formula

$$(\exists\mathsf{X}\mathbf{7}(\mathsf{X}\mathbf{7} \equiv \mathsf{X}\mathbf{6}) \rightarrow ((\mathsf{X}\mathbf{1} \equiv \mathsf{X}\mathbf{2}) \wedge \exists\mathsf{X}\mathbf{2}\text{Her}(\text{doli}, \text{doli}, \mathsf{X}\mathbf{2})))$$

tenemos que $Li(\varphi) = \{\mathsf{X}\mathbf{1}, \mathsf{X}\mathbf{2}, \mathsf{X}\mathbf{6}\}$ (justifique). Tambien si

$$\varphi = (\exists\mathsf{X}\mathbf{2}\text{Her}(\mathsf{X}\mathbf{2}, \mathsf{X}\mathbf{7}, \text{uno}) \rightarrow \text{Her}(\mathsf{X}\mathbf{2}, \mathsf{X}\mathbf{7}, \text{uno}))$$

entonces $Li(\varphi) = \{\mathsf{X}\mathbf{2}, \mathsf{X}\mathbf{7}\}$.

Una *sentencia* sera una formula φ tal que $Li(\varphi) = \emptyset$. Usaremos S^τ para denotar el conjunto de las sentencias de tipo τ.

Lemma 189 *(a)* $Li((t \equiv s)) = \{v \in Var : v \text{ ocurre en } t \text{ o } v \text{ ocurre en } s\}$.

(b) $Li(r(t_1, ..., t_n)) = \{v \in Var : v \text{ ocurre en algun } t_i\}$.

(c) $Li(\neg\varphi) = Li(\varphi)$

(d) $Li((\varphi\eta\psi)) = Li(\varphi) \cup Li(\psi)$.

(e) $Li(Qx_j\varphi) = Li(\varphi) - \{x_j\}$.

Proof. (a) y (b) son triviales de las definiciones y dejadas al lector

(d) Supongamos $v \in Li((\varphi\eta\psi))$, entonces hay un i tal que v ocurre libremente en $(\varphi\eta\psi)$ a partir de i. Por definicion tenemos que ya sea v ocurre libremente en φ a partir de $i-1$ o v ocurre libremente en ψ a partir de $i - |(\varphi\eta|$, con lo cual $v \in Li(\varphi) \cup Li(\psi)$

Supongamos ahora que $v \in Li(\varphi) \cup Li(\psi)$. S.p.d.g. supongamos $v \in Li(\psi)$. Por definicion tenemos que hay un i tal que v ocurre libremente en ψ a partir de i. Pero notese que esto nos dice por definicion que v ocurre libremente en $(\varphi\eta\psi)$ a partir de $i + |(\varphi\eta|$ con lo cual $v \in Li((\varphi\eta\psi))$.

(c) es similar a (d)

(e) Supongamos $v \in Li(Qx_j\varphi)$, entonces hay un i tal que v ocurre libremente en $Qx_j\varphi$ a partir de i. Por definicion tenemos que $v \neq x_j$ y v ocurre libremente en φ a partir de $i - |Qx_j|$, con lo cual $v \in Li(\varphi) - \{x_j\}$

Supongamos ahora que $v \in Li(\varphi) - \{x_j\}$. Por definicion tenemos que hay un i tal que v ocurre libremente en φ a partir de i. Ya que $v \neq x_j$ esto nos dice por definicion que v ocurre libremente en $Qx_j\varphi$ a partir de $i + |Qx_j|$, con lo cual $v \in Li(Qx_j\varphi)$. ■

6.7.4 Modelo matematico de las formulas elementales de tipo τ

Si $\tau = (\mathcal{C}, \mathcal{F}, \mathcal{R}, a)$ es un tipo, diremos que τ' es una *extension de* τ *por nombres de constante* si τ' es de la forma $(\mathcal{C}', \mathcal{F}, \mathcal{R}, a)$ con $\mathcal{C}'$ tal que $\mathcal{C} \subseteq \mathcal{C}'$.

Hemos definido las formulas de tipo τ con la intencion de dar un modelo matematico del concepto de formula elemental de tipo τ pero deberiamos notar que en las formulas de tipo τ no hay nombres de elementos fijos por lo cual dichas formulas son un modelo matematico solo de ciertas formulas elementales de tipo τ, a saber aquellas en las cuales no hay nombres de elementos fijos. Recordemos que estos nombres se usaban en las pruebas elementales para denotar elementos fijos (a veces arbitrarios y otras veces que cumplian alguna propiedad).

Cuando un matematico realiza una prueba elemental en una teoria elemental (Σ, τ) comienza la misma imaginando una estructura de tipo τ de la cual lo unico que sabe es que cumple las sentencias de Σ. Luego cuando fija un elemento y le pone nombre, digamos b, podemos pensar que expandio su estructura imaginaria a una de tipo $(\mathcal{C} \cup \{b\}, \mathcal{F}, \mathcal{R}, a)$ y continua su razonamiento. Esto lo puede hacer muchas veces a lo largo de una prueba por lo cual su estructura imaginaria va cambiando de tipo. Esta mecanica de prueba del matematico nos deja ver que es natural modelizar las formulas elementales de tipo τ con formulas de tipo τ_1, donde τ_1 es alguna extension de τ por constantes.

6.8 Modelo matematico del valor de verdad de una formula

En esta seccion daremos una definicion matematica que modeliza la idea intuitiva de cuando una formula de tipo τ es verdadera en una estructura dada para una asignacion de elementos a las variables libres de dicha formula. Esto corresponde al punto (2) del programa de logica enunciado en la Seccion 6.6.

6.8.1 El valor de un termino en una estructura

Sea $\mathbf{A} = (A, i)$ una estructura de tipo τ. Una *asignacion de* $\mathbf{A}$ sera un elemento de $A^{\mathbf{N}} = \{$infinituplas de elementos de $A\}$. Si $\vec{a} = (a_1, a_2, ...)$ es una asignacion, entonces diremos que a_j *es el valor que* $\vec{a}$ *le asigna a la variable* x_j.

Dada una estructura $\mathbf{A}$ de tipo τ, un termino $t \in T^\tau$ y una asignacion $\vec{a} = (a_1, a_2, ...) \in A^{\mathbf{N}}$ definamos recursivamente $t^{\mathbf{A}}[\vec{a}]$ de la siguiente manera

(1) Si $t = x_i \in Var$, entonces $t^{\mathbf{A}}[\vec{a}] = a_i$

(2) Si $t = c \in \mathcal{C}$, entonces $t^{\mathbf{A}}[\vec{a}] = i(c)$

(3) Si $t = f(t_1, ..., t_n)$, con $f \in \mathcal{F}_n$, $n \geq 1$ y $t_1, ..., t_n \in T^\tau$, entonces $t^{\mathbf{A}}[\vec{a}] = i(f)(t_1^{\mathbf{A}}[\vec{a}], ..., t_n^{\mathbf{A}}[\vec{a}])$

El elemento $t^{\mathbf{A}}[\vec{a}]$ sera llamado el *valor de* t *en la estructura* $\mathbf{A}$ *para la asignacion* $\vec{a}$.

Veamos un ejemplo. Sea τ el tipo

$$(\{\text{uno}, \text{doli}\}, \{\text{MAS}, \text{P}\}, \{\text{Her}\}, \{(\text{MAS}, 4), (\text{P}, 1), (\text{Her}, 3)\})$$

y sea $\mathbf{A} = (A, i)$ la estructura de tipo τ con universo $A = \mathbf{R}$ y

- $i(\text{uno}) = 9$

- $i(\text{doli}) = 0$

- $i(\text{MAS})$ igual a la operacion

$$\begin{array}{ccc} \mathbf{R}^4 & \rightarrow & \mathbf{R} \\ (x, y, z, w) & \rightarrow & 2x + 4y \end{array}$$

- $i(\text{P})$ igual a la operacion

$$\begin{array}{ccc} \mathbf{R} & \rightarrow & \mathbf{R} \\ x & \rightarrow & 5^x \end{array}$$

- $i(\text{Her}) = \{(x, y, z) \in \mathbf{R}^3 : x.y.z = 9\}$

Sea $\vec{a} = (1, 2, 3, 4, 5, ...)$. Claramente $\vec{a}$ es una asignacion de $\mathbf{A}$. Se tiene que:

1. Si $t = \mathsf{X}\mathit{55}\mathbf{4}$, entonces $t^{\mathbf{A}}[\vec{a}] = \mathsf{X}\mathit{55}\mathbf{4}^{\mathbf{A}}[\vec{a}] = 554$ (por (1) de la definicion recursiva de $t^{\mathbf{A}}[\vec{a}]$)

2. Si $t = \mathrm{uno}$, entonces $t^{\mathbf{A}}[\vec{a}] = \mathrm{uno}^{\mathbf{A}}[\vec{a}] = 9$ (por (2) de la definicion recursiva de $t^{\mathbf{A}}[\vec{a}]$)

3. Si $t = \mathrm{P}(\mathsf{X}\mathbf{3})$, entonces
$$\begin{aligned} t^{\mathbf{A}}[\vec{a}] &= \mathrm{P}(\mathsf{X}\mathbf{3})^{\mathbf{A}}[\vec{a}] \\ &= i(\mathrm{P})(\mathsf{X}\mathbf{3}^{\mathbf{A}}[\vec{a}]) \text{ (por (3) de la definicion de } t^{\mathbf{A}}[\vec{a}]) \\ &= i(\mathrm{P})(3) \\ &= 5^3 = 125 \end{aligned}$$

4. Si $t = \mathrm{MAS}(\mathsf{X}\mathbf{1}, \mathrm{uno}, \mathsf{X}\mathbf{3}, \mathsf{X}\mathit{55}\mathbf{4})$, entonces
$$\begin{aligned} t^{\mathbf{A}}[\vec{a}] &= \mathrm{MAS}(\mathsf{X}\mathbf{1}, \mathrm{uno}, \mathsf{X}\mathbf{3}, \mathsf{X}\mathit{55}\mathbf{4})^{\mathbf{A}}[\vec{a}] \\ &= i(\mathrm{MAS})(\mathsf{X}\mathbf{1}^{\mathbf{A}}[\vec{a}], \mathrm{uno}^{\mathbf{A}}[\vec{a}], \mathsf{X}\mathbf{3}^{\mathbf{A}}[\vec{a}], \mathsf{X}\mathit{55}\mathbf{4}^{\mathbf{A}}[\vec{a}]) \\ &= i(\mathrm{MAS})(1, 9, 3, 554) \\ &= 2.1 + 4.9 = 38 \end{aligned}$$

Lemma 190 *Sea* $\mathbf{A}$ *una estructura de tipo* τ *y sea* $t \in T^{\tau}$. *Supongamos que* $\vec{a}, \vec{b}$ *son asignaciones tales que* $a_i = b_i$, *cada vez que* x_i *ocurra en* t. *Entonces* $t^{\mathbf{A}}[\vec{a}] = t^{\mathbf{A}}[\vec{b}]$.

Proof. Sea

- Teo_k: El lema vale para $t \in T_k^{\tau}$.

Teo_0 es facil de probar. Veamos $\mathrm{Teo}_k \Rightarrow \mathrm{Teo}_{k+1}$. Supongamos $t \in T_{k+1}^{\tau} - T_k^{\tau}$ y sean $\vec{a}, \vec{b}$ asignaciones tales que $a_i = b_i$, cada vez que x_i ocurra en t. Notese que $t = f(t_1, ..., t_n)$, con $f \in \mathcal{F}_n$, $n \geq 1$ y $t_1, ..., t_n \in T_k^{\tau}$. Notese que para cada $j = 1, ..., n$, tenemos que $a_i = b_i$, cada vez que x_i ocurra en t_j, lo cual por Teo_k nos dice que
$$t_j^{\mathbf{A}}[\vec{a}] = t_j^{\mathbf{A}}[\vec{b}],\ j = 1, ..., n$$
Se tiene entonces que
$$\begin{aligned} t^{\mathbf{A}}[\vec{a}] &= i(f)(t_1^{\mathbf{A}}[\vec{a}], ..., t_n^{\mathbf{A}}[\vec{a}]) \text{ (por def de } t^{\mathbf{A}}[\vec{a}]) \\ &= i(f)(t_1^{\mathbf{A}}[\vec{b}], ..., t_n^{\mathbf{A}}[\vec{b}]) \\ &= t^{\mathbf{A}}[\vec{b}] \text{ (por def de } t^{\mathbf{A}}[\vec{b}]) \end{aligned}$$

■

6.8.2 La relacion $\models$

Dada una asignacion $\vec{a} \in A^{\mathbf{N}}$ y $a \in A$, con $\downarrow_i^a(\vec{a})$ denotaremos la asignacion que resulta de reemplazar en $\vec{a}$ el i-esimo elemento por a. A continuacion definiremos recursivamente la relacion $\mathbf{A} \models \varphi[\vec{a}]$, donde $\mathbf{A}$ es una estructura de tipo τ, $\vec{a}$ es una asignacion y $\varphi \in F^{\tau}$. Escribiremos $\mathbf{A} \not\models \varphi[\vec{a}]$ para expresar que no se da $\mathbf{A} \models \varphi[\vec{a}]$.

(1) Si $\varphi = (t \equiv s)$, entonces

- $\mathbf{A} \models \varphi[\vec{a}]$ si y solo si $t^{\mathbf{A}}[\vec{a}] = s^{\mathbf{A}}[\vec{a}]$

(2) Si $\varphi = r(t_1, ..., t_m)$, entonces

- $\mathbf{A} \models \varphi[\vec{a}]$ si y solo si $(t_1^{\mathbf{A}}[\vec{a}], ..., t_m^{\mathbf{A}}[\vec{a}]) \in i(r)$

(3) Si $\varphi = (\varphi_1 \wedge \varphi_2)$, entonces

- $\mathbf{A} \models \varphi[\vec{a}]$ si y solo si $\mathbf{A} \models \varphi_1[\vec{a}]$ y $\mathbf{A} \models \varphi_2[\vec{a}]$

(4) Si $\varphi = (\varphi_1 \vee \varphi_2)$, entonces

- $\mathbf{A} \models \varphi[\vec{a}]$ si y solo si $\mathbf{A} \models \varphi_1[\vec{a}]$ o $\mathbf{A} \models \varphi_2[\vec{a}]$

(5) Si $\varphi = (\varphi_1 \rightarrow \varphi_2)$, entonces

- $\mathbf{A} \models \varphi[\vec{a}]$ si y solo si $\mathbf{A} \not\models \varphi_1[\vec{a}]$ o $\mathbf{A} \models \varphi_2[\vec{a}]$

(6) Si $\varphi = (\varphi_1 \leftrightarrow \varphi_2)$, entonces

- $\mathbf{A} \models \varphi[\vec{a}]$ si y solo si ya sea se dan $\mathbf{A} \models \varphi_1[\vec{a}]$ y $\mathbf{A} \models \varphi_2[\vec{a}]$ o se dan $\mathbf{A} \not\models \varphi_1[\vec{a}]$ y $\mathbf{A} \not\models \varphi_2[\vec{a}]$

(7) Si $\varphi = \neg\varphi_1$, entonces

- $\mathbf{A} \models \varphi[\vec{a}]$ si y solo si $\mathbf{A} \not\models \varphi_1[\vec{a}]$

(8) Si $\varphi = \forall x_i \varphi_1$, entonces

- $\mathbf{A} \models \varphi[\vec{a}]$ si y solo si para cada $a \in A$, se da que $\mathbf{A} \models \varphi_1[\downarrow_i^a(\vec{a})]$

(9) Si $\varphi = \exists x_i \varphi_1$, entonces

- $\mathbf{A} \models \varphi[\vec{a}]$ si y solo si hay un $a \in A$ tal que $\mathbf{A} \models \varphi_1[\downarrow_i^a(\vec{a})]$

Cuando se de $\mathbf{A} \models \varphi[\vec{a}]$ diremos que *la estructura* $\mathbf{A}$ *satisface* φ *en la asignacion* $\vec{a}$ y en tal caso diremos que φ *es verdadera en* $\mathbf{A}$ *para la asignacion* $\vec{a}$. Cuando no se de $\mathbf{A} \models \varphi[\vec{a}]$ diremos que *la estructura* $\mathbf{A}$ *no satisface* φ *en la asignacion* $\vec{a}$ y en tal caso diremos que φ *es falsa en* $\mathbf{A}$ *para la asignacion* $\vec{a}$. Tambien hablaremos del *valor de verdad de* φ *en* $\mathbf{A}$ *para la asignacion* $\vec{a}$ el cual sera igual a 1 si se da $\mathbf{A} \models \varphi[\vec{a}]$ y 0 en caso contrario.

Veamos algunos ejemplos. Sea τ el tipo

$$(\{\text{uno}, \text{doli}\}, \{\text{MAS}, \text{P}\}, \{\text{Her}\}, \{(\text{MAS}, 4), (\text{P}, 1), (\text{Her}, 3)\})$$

y sea $\mathbf{A} = (A, i)$ la estructura de tipo τ con universo $A = \mathbf{R}$ y

- $i(\text{uno}) = 9$

- $i(\text{doli}) = 0$

- $i(\text{MAS})$ igual a la operacion

$$\begin{array}{ccc} \mathbf{R}^4 & \rightarrow & \mathbf{R} \\ (x, y, z, w) & \rightarrow & 2x + 4y \end{array}$$

- $i(\text{P})$ igual a la operacion

$$\begin{array}{ccc} \mathbf{R} & \rightarrow & \mathbf{R} \\ x & \rightarrow & 5^x \end{array}$$

- $i(\text{Her}) = \{(x, y, z) \in \mathbf{R}^3 : x.y.z = 9\}$

Sea $\vec{a} = (1, 2, 3, 4, 5, ...)$. Claramente $\vec{a}$ es una asignacion de $\mathbf{A}$. Consideremos los siguientes ejemplos:

(E1) Si $\varphi = (\text{MAS}(\mathsf{X}\mathbf{1}, \text{uno}, \mathsf{X}\mathbf{3}, \mathsf{X}\mathit{55}\mathbf{4}) \equiv \text{P}(\mathsf{X}\mathbf{3}))$, entonces ya que

(a) $\text{MAS}(\mathsf{X}\mathbf{1}, \text{uno}, \mathsf{X}\mathbf{3}, \mathsf{X}\mathit{55}\mathbf{4})^{\mathbf{A}}[\vec{a}] = 38$

(b) $\text{P}(\mathsf{X}\mathbf{3})^{\mathbf{A}}[\vec{a}] = 125$

tenemos que (1) de la definicion nos dice que $\mathbf{A} \models \varphi[\vec{a}]$ si y solo si $38 = 125$ por lo cual se saca que $\mathbf{A} \not\models \varphi[\vec{a}]$.

(E2) Si $\varphi = \neg\text{Her}(\text{P}(\text{P}(\mathsf{X}\mathbf{6})), \mathsf{X}\mathbf{3}, \text{doli})$, entonces ya que

- $\text{P}(\text{P}(\mathsf{X}\mathbf{6}))^{\mathbf{A}}[\vec{a}] = 5^{(5^6)}$
- $\mathsf{X}\mathbf{3}^{\mathbf{A}}[\vec{a}] = 3$
- $\text{doli}^{\mathbf{A}}[\vec{a}] = 0$

tenemos que (7) de la definicion nos dice que $\mathbf{A} \models \varphi[\vec{a}]$ si y solo si $\mathbf{A} \not\models \text{Her}(\text{P}(\text{P}(\mathsf{X}\mathbf{6})), \mathsf{X}\mathbf{3}, \text{doli})[\vec{a}]$. Pero (2) de la definicion nos dice que $\mathbf{A} \models \text{Her}(\text{P}(\text{P}(\mathsf{X}\mathbf{6})), \mathsf{X}\mathbf{3}, \text{doli})[\vec{a}]$ si y solo si $(5^{(5^6)}, 3, 0) \in i(\text{Her})$ ya que no se da que $(5^{(5^6)}, 3, 0) \in i(\text{Her})$, tenemos que $\mathbf{A} \not\models \text{Her}(\text{P}(\text{P}(\mathsf{X}\mathbf{6})), \mathsf{X}\mathbf{3}, \text{doli})[\vec{a}]$ lo cual nos dice que $\mathbf{A} \models \varphi[\vec{a}]$.

(E3) Si $\varphi = \exists \mathsf{X}\mathbf{3}\mathrm{Her}(\mathsf{X}\mathbf{6}, \mathsf{X}\mathbf{3}, \mathrm{uno})$, entonces por (9) de la definicion tenemos que

- $\mathbf{A} \models \varphi[\vec{a}]$ sii hay un $r \in \mathbf{R}$ tal que $\mathbf{A} \models \mathrm{Her}(\mathsf{X}\mathbf{6}, \mathsf{X}\mathbf{3}, \mathrm{uno})[\downarrow_3^r (\vec{a})]$

es decir que

- $\mathbf{A} \models \varphi[\vec{a}]$ sii hay un $r \in \mathbf{R}$ tal que $\mathbf{A} \models \mathrm{Her}(\mathsf{X}\mathbf{6}, \mathsf{X}\mathbf{3}, \mathrm{uno})[(1, 2, r, 4, 5, 6, ...)]$

Pero (2) de la definicion nos dice que cualquiera sea $r \in \mathbf{R}$ se tiene que

- $\mathbf{A} \models \mathrm{Her}(\mathsf{X}\mathbf{6}, \mathsf{X}\mathbf{3}, \mathrm{uno})[(1, 2, r, 4, 5, 6, ...)]$ sii $(6, r, 9) \in i(\mathrm{Her})$

O sea que obtenemos finalmente que

- $\mathbf{A} \models \varphi[\vec{a}]$ sii hay un $r \in \mathbf{R}$ tal que $6.r.9 = 9$

Lo cual claramente implica que $\mathbf{A} \models \varphi[\vec{a}]$ ya que podemos tomar $r = 1/6$.

(E4) Si $\varphi = \forall \mathsf{X}\mathbf{3}((\mathsf{X}\mathbf{4} \equiv \mathsf{X}\mathbf{3}) \rightarrow \exists \mathsf{X}\mathbf{6}\mathrm{Her}(\mathsf{X}\mathbf{6}, \mathsf{X}\mathbf{3}, \mathrm{uno}))$, entonces por (8) de la definicion tenemos que

- $\mathbf{A} \models \varphi[\vec{a}]$ sii para cada $r \in \mathbf{R}$ se da que

$$\mathbf{A} \models ((\mathsf{X}\mathbf{4} \equiv \mathsf{X}\mathbf{3}) \rightarrow \exists \mathsf{X}\mathbf{6}\mathrm{Her}(\mathsf{X}\mathbf{6}, \mathsf{X}\mathbf{3}, \mathrm{uno}))[\downarrow_3^r (\vec{a})]$$

es decir que

- $\mathbf{A} \models \varphi[\vec{a}]$ sii para cada $r \in \mathbf{R}$ se da que

$$\mathbf{A} \models ((\mathsf{X}\mathbf{4} \equiv \mathsf{X}\mathbf{3}) \rightarrow \exists \mathsf{X}\mathbf{6}\mathrm{Her}(\mathsf{X}\mathbf{6}, \mathsf{X}\mathbf{3}, \mathrm{uno}))[(1, 2, r, 4, 5, 6, ...)]$$

Pero entonces (5) de la de∇finicion nos dice que

- $\mathbf{A} \models \varphi[\vec{a}]$ sii para cada $r \in \mathbf{R}$ se da que

  $\mathbf{A} \not\models (\mathsf{X}\mathbf{4} \equiv \mathsf{X}\mathbf{3})[(1, 2, r, 4, 5, 6, ...)]$ o $\mathbf{A} \models \exists \mathsf{X}\mathbf{6}\mathrm{Her}(\mathsf{X}\mathbf{6}, \mathsf{X}\mathbf{3}, \mathrm{uno}))[(1, 2, r, 4, 5, 6, ...)]$

  O sea que

- $\mathbf{A} \models \varphi[\vec{a}]$ sii para cada $r \in \mathbf{R}$ se da que

$$r \neq 4 \text{ o } \mathbf{A} \models \exists \mathsf{X}\mathbf{6}\mathrm{Her}(\mathsf{X}\mathbf{6}, \mathsf{X}\mathbf{3}, \mathrm{uno}))[(1, 2, r, 4, 5, 6, ...)]$$

  Es decir que debemos ver cuando se da que $\mathbf{A} \models \exists \mathsf{X}\mathbf{6}\mathrm{Her}(\mathsf{X}\mathbf{6}, \mathsf{X}\mathbf{3}, \mathrm{uno}))[(1, 2, r, 4, 5, 6, ...)]$. Por (9) y (2) de la definicion tenemos que cualquiera sea el $r \in \mathbf{R}$ se da que

- $\mathbf{A} \models \exists \mathsf{X}\mathbf{6}\mathrm{Her}(\mathsf{X}\mathbf{6}, \mathsf{X}\mathbf{3}, \mathrm{uno}))[(1, 2, r, 4, 5, 6, ...)]$ sii hay un $s \in \mathbf{R}$ tal que $s.r.9 = 9$.

Esto nos dice finalmente que

- $\mathbf{A} \models \varphi[\vec{a}]$ sii para cada $r \in \mathbf{R}$ se da que

$$r \neq 4 \text{ o hay un } s \in \mathbf{R} \text{ tal que } s.r.9 = 9$$

Pensando un poco esto nos dice que $\mathbf{A} \models \varphi[\vec{a}]$ (separar los casos $r = 4$ y $r \neq 4$)

Lemma 191 *Supongamos que $\vec{a}, \vec{b}$ son asignaciones tales que si $x_i \in Li(\varphi)$, entonces $a_i = b_i$. Entonces $\mathbf{A} \models \varphi[\vec{a}]$ sii $\mathbf{A} \models \varphi[\vec{b}]$*

Proof. Probaremos por induccion en k que el lema vale para cada $\varphi \in F_k^\tau$. El caso $k = 0$ se desprende del Lema 190. Veamos que Teo_k implica Teo_{k+1}. Sea $\varphi \in F_{k+1}^\tau - F_k^\tau$. Hay varios casos:

CASO $\varphi = (\varphi_1 \wedge \varphi_2)$.

Ya que $Li(\varphi_i) \subseteq Li(\varphi)$, $i = 1, 2$, Teo_k nos dice que $\mathbf{A} \models \varphi_i[\vec{a}]$ sii $\mathbf{A} \models \varphi_i[\vec{b}]$, para $i = 1, 2$. Se tiene entonces que

$$\begin{array}{l}
\mathbf{A} \models \varphi[\vec{a}] \\
\ \Updownarrow \ \text{(por (3) en la def de } \mathbf{A} \models \varphi[\vec{a}]) \\
\mathbf{A} \models \varphi_1[\vec{a}] \text{ y } \mathbf{A} \models \varphi_2[\vec{a}] \\
\ \Updownarrow \ (\text{por Teo}_k) \\
\mathbf{A} \models \varphi_1[\vec{b}] \text{ y } \mathbf{A} \models \varphi_2[\vec{b}] \\
\ \Updownarrow \ \text{(por (3) en la def de } \mathbf{A} \models \varphi[\vec{a}]) \\
\mathbf{A} \models \varphi[\vec{b}]
\end{array}$$

CASO $\varphi = (\varphi_1 \vee \varphi_2)$.

Es completamente similar al anterior.

CASO $\varphi = (\varphi_1 \rightarrow \varphi_2)$.

Es completamente similar al anterior.

CASO $\varphi = (\varphi_1 \leftrightarrow \varphi_2)$.

Es completamente similar al anterior.

CASO $\varphi = \neg\varphi_1$.

Es completamente similar al anterior.

CASO $\varphi = \forall x_j \varphi_1$.

Supongamos $\mathbf{A} \models \varphi[\vec{a}]$. Entonces por (8) en la def de $\mathbf{A} \models \varphi[\vec{a}]$ se tiene que $\mathbf{A} \models \varphi_1[\downarrow_j^a (\vec{a})]$, para todo $a \in A$. Notese que $\downarrow_j^a (\vec{a})$ y $\downarrow_j^a (\vec{b})$ coinciden en toda x_i de $x_i \in Li(\varphi_1) \subseteq Li(\varphi_1) \cup \{x_j\}$, con lo cual por Teo_k se tiene que $\mathbf{A} \models \varphi_1[\downarrow_j^a (\vec{b})]$, para todo $a \in A$, lo cual por (8) en la def de $\mathbf{A} \models \varphi[\vec{a}]$ nos dice que $\mathbf{A} \models \varphi[\vec{b}]$. La prueba de que $\mathbf{A} \models \varphi[\vec{b}]$ implica que $\mathbf{A} \models \varphi[\vec{a}]$ es similar.

CASO $\varphi = \exists x_j \varphi_1$.

Es similar al anterior. $\blacksquare$

Corollary 192 *Si φ es una sentencia, entonces $\mathbf{A} \models \varphi[\vec{a}]$ sii $\mathbf{A} \models \varphi[\vec{b}]$, cualesquiera sean las asignaciones $\vec{a}, \vec{b}$.*

En virtud del corolario anterior tenemos que el valor de verdad de una sentencia φ en una estructura dada $\mathbf{A}$ para una asignacion $\vec{a}$ no depende de $\vec{a}$, es decir este valor es ya sea 1 para todas las asignaciones o 0 para todas las asignaciones. En el primer caso diremos que φ *es verdadera en* $\mathbf{A}$ (y escribiremos $\mathbf{A} \models \varphi$) y en el segundo caso diremos que φ *es falsa en* $\mathbf{A}$ (y escribiremos $\mathbf{A} \not\models \varphi$)

Una sentencia de tipo τ sera llamada *universalmente valida* si es verdadera en cada modelo de tipo τ.

6.8.3 Equivalencia de formulas

Dadas $\varphi, \psi \in F^\tau$ diremos que φ y ψ son *equivalentes* cuando se de la siguiente condicion

- $\mathbf{A} \models \varphi[\vec{a}]$ si y solo si $\mathbf{A} \models \psi[\vec{a}]$, para cada modelo de tipo τ, $\mathbf{A}$ y cada $\vec{a} \in A^{\mathbf{N}}$

Escribiremos $\varphi \sim \psi$ cuando φ y ψ sean equivalentes. Notese que

$$\{(\varphi, \psi) \in F^\tau : \varphi \sim \psi\}$$

es una relacion de equivalencia sobre F^τ.

Lemma 193 *(a) Si $Li(\phi) \cup Li(\psi) \subseteq \{x_{i_1}, ..., x_{i_n}\}$, entonces $\phi \sim \psi$ si y solo si la sentencia $\forall x_{i_1} ... \forall x_{i_n} (\phi \leftrightarrow \psi)$ es universalmente valida.*

(b) Si $\phi_i \sim \psi_i$, $i = 1, 2$, entonces $\neg\phi_1 \sim \neg\psi_1$, $(\phi_1 \eta \phi_2) \sim (\psi_1 \eta \psi_2)$ y $Qv\phi_1 \sim Qv\psi_1$.

(c) Si $\phi \sim \psi$ y α' es el resultado de reemplazar en una formula α algunas (posiblemente 0) ocurrencias de ϕ por ψ, entonces $\alpha \sim \alpha'$.

Proof. (a) Tenemos que

$$\begin{array}{l}
\gamma \sim \psi \\
\ \Updownarrow \ \text{(por (6) de la def de} \models \text{)} \\
\mathbf{A} \models (\gamma \leftrightarrow \psi)[\vec{a}], \text{ para todo } \mathbf{A} \text{ y toda } \vec{a} \in A^{\mathbf{N}} \\
\ \Updownarrow \\
\mathbf{A} \models (\gamma \leftrightarrow \psi)[\downarrow_{i_n}^{a}(\vec{a})], \text{ para todo } \mathbf{A},\ a \in A \text{ y toda } \vec{a} \in A^{\mathbf{N}} \\
\ \Updownarrow \ \text{(por (8) de la def de} \models \text{)} \\
\mathbf{A} \models \forall x_{i_n}(\gamma \leftrightarrow \psi)[\vec{a}], \text{ para todo } \mathbf{A} \text{ y toda } \vec{a} \in A^{\mathbf{N}} \\
\ \Updownarrow \\
\mathbf{A} \models \forall x_{i_n}(\gamma \leftrightarrow \psi)[\downarrow_{i_{n-1}}^{a}(\vec{a})], \text{ para todo } \mathbf{A},\ a \in A \text{ y toda } \vec{a} \in A^{\mathbf{N}} \\
\ \Updownarrow \ \text{(por (8) de la def de} \models \text{)} \\
\mathbf{A} \models \forall x_{i_{n-1}} \forall x_{i_n}(\gamma \leftrightarrow \psi)[\vec{a}], \text{ para todo } \mathbf{A} \text{ y toda } \vec{a} \in A^{\mathbf{N}} \\
\ \Updownarrow \\
\quad \vdots \\
\ \Updownarrow \\
\mathbf{A} \models \forall x_{i_1} ... \forall x_{i_n}(\gamma \leftrightarrow \psi)[\vec{a}], \text{ para todo } \mathbf{A} \text{ y toda } \vec{a} \in A^{\mathbf{N}} \\
\ \Updownarrow \\
\forall x_{i_1} ... \forall x_{i_n}(\gamma \leftrightarrow \psi) \text{ es universalmente valida}
\end{array}$$

(b) Es dejado al lector.
(c) Por induccion en el k tal que $\alpha \in F_k^\tau$. ■

6.9 Un poco de semantica

Dado que las estructuras de tipo τ constituyen los "mundos posibles" donde las formulas de tipo τ se "interpretan" se suele llamar semantica al estudio general de las estructuras y su vinculacion con el lenguaje. Aqui daremos algunas nociones basicas de semantica.

6.9.1 Homomorfismos

La nocion de homomorfismo estaba restringida a unos pocos casos particulares de estructuras estudiadas pero ahora con nuestra definicion general de estructura podemos generalizarla en forma natural. Antes una notacion muy util. Dado un modelo de tipo τ, $\mathbf{A} = (A, i)$, para cada $s \in \mathcal{C} \cup \mathcal{F} \cup \mathcal{R}$, usaremos $s^{\mathbf{A}}$ para denotar a $i(s)$.

Sean $\mathbf{A}$ y $\mathbf{B}$ modelos de tipo τ. Una funcion $F : A \to B$ sera un *homomorfismo de* $\mathbf{A}$ *en* $\mathbf{B}$ si se cumplen las siguientes

(1) $F(c^{\mathbf{A}}) = c^{\mathbf{B}}$, para todo $c \in \mathcal{C}$,

(2) $F(f^{\mathbf{A}}(a_1, ..., a_n)) = f^{\mathbf{B}}(F(a_1), ..., F(a_n))$, para cada $f \in \mathcal{F}_n$, $a_1, ..., a_n \in A$.

(3) $(a_1, ..., a_n) \in r^{\mathbf{A}}$ implica $(F(a_1), ..., F(a_n)) \in r^{\mathbf{B}}$, para todo $r \in \mathcal{R}_n$, $a_1, ..., a_n \in A$.

Un *isomorfismo de* $\mathbf{A}$ *en* $\mathbf{B}$ sera un un homomorfismo de $\mathbf{A}$ en $\mathbf{B}$ el cual sea biyectivo y cuya inversa sea un homomorfismo de $\mathbf{B}$ en $\mathbf{A}$. Diremos que los modelos $\mathbf{A}$ y $\mathbf{B}$ son *isomorfos* (en simbolos: $\mathbf{A} \cong \mathbf{B}$), cuando haya un isomorfismo F de $\mathbf{A}$ en $\mathbf{B}$. Diremos que $F : \mathbf{A} \to \mathbf{B}$ *es un homomorfismo* para expresar que F es un homomorfismo de $\mathbf{A}$ en $\mathbf{B}$. Analogamente diremos que $F : \mathbf{A} \to \mathbf{B}$ *es un isomorfismo* para expresar que F es un isomorfismo de $\mathbf{A}$ en $\mathbf{B}$.

Ejercicio: Pruebe que la relacion $\cong$ es reflexiva, transitiva y simetrica.

Lemma 194 *Sea $F : \mathbf{A} \to \mathbf{B}$ un homomorfismo. Entonces*

$$F(t^{\mathbf{A}}[(a_1, a_2, ...)] = t^{\mathbf{B}}[(F(a_1), F(a_2), ...)]$$

para cada $t \in T^\tau$, $(a_1, a_2, ...) \in A^{\mathbf{N}}$.

Proof. Por induccion. Sea

- Teo$_k$: Si $F : \mathbf{A} \to \mathbf{B}$ es un homomorfismo, entonces

$$F(t^{\mathbf{A}}[(a_1, a_2, ...)] = t^{\mathbf{B}}[(F(a_1), F(a_2), ...)]$$

para cada $t \in T_k^\tau$, $(a_1, a_2, ...) \in A^{\mathbf{N}}$.

Teo$_0$ es trivial. Veamos que Teo$_k$ implica Teo$_{k+1}$. Supongamos que vale Teo$_k$ y supongamos $F : \mathbf{A} \to \mathbf{B}$ es un homomorfismo, $t \in T_{k+1}^\tau - T_k^\tau$ y $\vec{a} = (a_1, a_2, ...) \in A^{\mathbf{N}}$. Denotemos $(F(a_1), F(a_2), ...)$ con $F(\vec{a})$. Por Lema 175, $t = f(t_1, ..., t_n)$, con $n \geq 1$, $f \in \mathcal{F}_n$ y $t_1, ..., t_n \in T_k^\tau$. Tenemos entonces

$$\begin{array}{rcl}
F(t^{\mathbf{A}}[\vec{a}]) & = & F(f(t_1, ..., t_n)^{\mathbf{A}}[\vec{a}]) \\
& = & F(f^{\mathbf{A}}(t_1^{\mathbf{A}}[\vec{a}], ..., t_n^{\mathbf{A}}[\vec{a}])) \\
& = & f^{\mathbf{B}}(F(t_1^{\mathbf{A}}[\vec{a}]), ..., F(t_n^{\mathbf{A}}[\vec{a}])) \\
& = & f^{\mathbf{B}}(t_1^{\mathbf{B}}[F(\vec{a})], ..., t_n^{\mathbf{B}}[F(\vec{a})])) \\
& = & f(t_1, ..., t_n)^{\mathbf{B}}[F(\vec{a})] \\
& = & t^{\mathbf{B}}[F(\vec{a})]
\end{array}$$

■

Lemma 195 *Supongamos que $F : \mathbf{A} \to \mathbf{B}$ es un isomorfismo. Sea $\varphi \in F^\tau$. Entonces*

$$\mathbf{A} \models \varphi[(a_1, a_2, ...)] \textit{ sii } \mathbf{B} \models \varphi[(F(a_1), F(a_2), ...)]$$

para cada $(a_1, a_2, ...) \in A^{\mathbf{N}}$. En particular $\mathbf{A}$ y $\mathbf{B}$ satisfacen las mismas sentencias de tipo τ.

Proof. Por induccion. Sea

- Teo$_k$: Supongamos que $F : \mathbf{A} \rightarrow \mathbf{B}$ es un isomorfismo. Sea $\varphi \in F_k^\tau$. Entonces

$$\mathbf{A} \models \varphi[(a_1, a_2, ...)] \text{ sii } \mathbf{B} \models \varphi[(F(a_1), F(a_2), ...)]$$

para cada $(a_1, a_2, ...) \in A^{\mathbf{N}}$

Prueba de Teo$_0$. Supongamos que $F : \mathbf{A} \rightarrow \mathbf{B}$ es un isomorfismo, $\varphi \in F_0^\tau$ y $(a_1, a_2, ...) \in A^{\mathbf{N}}$. Probaremos que

$$\mathbf{A} \models \varphi[(a_1, a_2, ...)] \text{ sii } \mathbf{B} \models \varphi[(F(a_1), F(a_2), ...)]$$

Hay dos casos. Caso $\varphi = r(t_1, ..., t_n)$, con $n \geq 1$, $r \in \mathcal{R}_n$ y $t_1, ..., t_n \in T^\tau$. Denotemos con $\vec{a}$ a $(a_1, a_2, ...)$ y con $F(\vec{a})$ a $(F(a_1), F(a_2), ...)$. Tenemos entonces

$$\begin{array}{rll} \mathbf{A} \models \varphi[\vec{a}] & \text{sii} & (t_1^{\mathbf{A}}[\vec{a}], ..., t_m^{\mathbf{A}}[\vec{a}]) \in r^{\mathbf{A}} \text{ (def de } \models \text{)} \\ & \text{sii} & (F(t_1^{\mathbf{A}}[\vec{a}]), ..., F(t_n^{\mathbf{A}}[\vec{a}])) \in r^{\mathbf{B}} \text{ (} F \text{ es iso)} \\ & \text{sii} & (t_1^{\mathbf{B}}[F(\vec{a})]), ..., t_n^{\mathbf{B}}[F(\vec{a})]) \in r^{\mathbf{B}} \text{ (Lema 194)} \\ & \text{sii} & \mathbf{B} \models \varphi[F(\vec{a})] \end{array}$$

Dejamos al lector completar la prueba de que Teo$_k$ implica Teo$_{k+1}$ ■

6.9.2 Algebras

Un tipo τ sera llamado *algebraico* si no contiene nombres de relacion (i.e. $\mathcal{R} = \emptyset$). Un modelo de un tipo algebraico τ sera llamado una τ-*algebra*. Ejemplos clasicos de τ-algebras son los grupos ($\tau = (\{e\}, \{.^2\}, \emptyset, a)$), los reticulados, los reticulados acotados, las algebras de Boole, etc. Muchos de los resultados y definiciones dados en la Seccion 4 para reticulados terna, reticulados acotados y reticulados complementados pueden ahora ser generalizados naturalmente para τ-algebras. Desarrollaremos un poco esta linea de "algebra general" la cual ha tenido un furte impacto en el area de las especificaciones algebraicas de tipos de datos.

Lemma 196 *Supongamos τ es algebraico. Si $F : \mathbf{A} \rightarrow \mathbf{B}$ es un homomorfismo biyectivo, entonces F es un isomorfismo.*

Proof. Solo falta probar que F^{-1} es un homomorfismo. Supongamos que $c \in \mathcal{C}$. Ya que $F(c^{\mathbf{A}}) = c^{\mathbf{B}}$, tenemos que $F^{-1}(c^{\mathbf{B}}) = c^{\mathbf{A}}$, por lo cual F^{-1} cumple (1) de la definicion de homomorfismo. Supongamos ahora que $f \in \mathcal{F}_n$ y sean $b_1, ..., b_n \in B$. Sean $a_1, ..., a_n \in A$ tales que $F(a_i) = b_i$, $i = 1, ..., n$. Tenemos que

$$\begin{array}{rcl} F^{-1}(f^{\mathbf{B}}(b_1, ..., b_n)) & = & F^{-1}(f^{\mathbf{B}}(F(a_1), ..., F(a_n))) \\ & = & F^{-1}(F(f^{\mathbf{A}}(a_1, ..., a_n))) \\ & = & f^{\mathbf{A}}(a_1, ..., a_n) \\ & = & f^{\mathbf{A}}(F^{-1}(b_1), ..., F^{-1}(b_n)) \end{array}$$

por lo cual F^{-1} satisface (2) de la definicion de homomorfismo ■

Subalgebras Dadas τ-algebras $\mathbf{A}$ y $\mathbf{B}$, diremos que $\mathbf{A}$ es una *subalgebra de* $\mathbf{B}$ cuando se den las siguientes condiciones

(1) $A \subseteq B$

(2) $c^{\mathbf{A}} = c^{\mathbf{B}}$, para cada $c \in \mathcal{C}$

(3) $f^{\mathbf{A}} = f^{\mathbf{B}}|_{A^n}$, para cada $f \in \mathcal{F}_n$, $n \geq 1$

Si $\mathbf{B}$ es una τ-algebra, entonces un *subuniverso de* $\mathbf{B}$ es un conjunto A el cual cumple las siguientes condiciones:

(1) $\emptyset \neq A \subseteq B$

(2) $c^{\mathbf{B}} \in A$, para cada $c \in \mathcal{C}$

(3) $f^{\mathbf{B}}(a_1, ..., a_n) \in A$, para cada $(a_1, ..., a_n) \in A^n$, $f \in \mathcal{F}_n$

Es importante notar que si bien los conceptos de subalgebra y subuniverso estan muy relacionados, se trata de objetos diferentes ya que las subalgebras de un algebra dada son estructuras de tipo τ y por lo tanto son pares ordenados y los subuniversos de un algebra dada son ciertos subconjuntos por lo cual no son pares ordenados. A continuacion presisaremos la relacion que hay entre estos dos conceptos. Notese que dado un subuniverso A de una τ-algebra $\mathbf{B}$ podemos definir en forma natural una τ-algebra $\mathbf{A}$ de la siguiente manera:

(1) Universo de $\mathbf{A} = A$

(2) $c^{\mathbf{A}} = c^{\mathbf{B}}$, para cada $c \in \mathcal{C}$

(3) $f^{\mathbf{A}} = f^{\mathbf{B}}|_{A^n}$, para cada $f \in \mathcal{F}_n$.

Es facil chequear que el algebra $\mathbf{A}$ asi definida es una subalgebra de $\mathbf{B}$. Lo anterior nos muestra que los subuniversos de un algebra dada son precisamente los universos de las distintas subalgebras de dicha algebra.

Lemma 197 *Supongamos τ es algebraico. Si $F : \mathbf{A} \to \mathbf{B}$ es un homomorfismo, entonces I_F es un subuniverso de* $\mathbf{B}$

Proof. Ya que $A \neq \emptyset$, tenemos que $I_F \neq \emptyset$. Es claro que $c^{\mathbf{B}} = F(c^{\mathbf{A}}) \in I_F$, para cada $c \in \mathcal{C}$. Sea $f \in \mathcal{F}_n$ y sean $b_1, ..., b_n \in I_F$ Sean $a_1, ..., a_n$ tales que $F(a_i) = b_i$, $i = 1, ..., n$. Tenemos que

$$f^{\mathbf{B}}(b_1, ..., b_n) = f^{\mathbf{B}}(F(a_1), ..., F(a_n)) = F(f^{\mathbf{A}}(a_1, ..., a_n)) \in I_F$$

por lo cual I_F es cerrada bajo $f^{\mathbf{B}}$. ■

Congruencias Sea $\mathbf{A}$ una τ-algebra. Una *congruencia sobre* $\mathbf{A}$ es una relacion de equivalencia θ sobre A la cual cumple que

$$a_1\theta b_1, ..., a_n\theta b_n \text{ implica } f^{\mathbf{A}}(a_1, ..., a_n)\theta f^{\mathbf{A}}(b_1, ..., b_n)$$

cualesquiera sean $a_1, ..., a_n, b_1, ..., b_n \in A$ y $f \in \mathcal{F}_n$.

Dada una congruencia θ sobre $\mathbf{A}$ se puede formar una nueva algebra $\mathbf{A}/\theta$ de la siguiente manera:

(1) Universo de $\mathbf{A}/\theta = A/\theta = \{a/\theta : a \in A\} = \{\text{clases de equivalencia de } \theta\}$

(2) $f^{\mathbf{A}/\theta}(a_1/\theta, ..., a_n/\theta) = f^{\mathbf{A}}(a_1, ..., a_n)/\theta$, para cada $f \in \mathcal{F}_n$.

(3) $c^{\mathbf{A}/\theta} = c^{\mathbf{A}}/\theta$, para cada $c \in \mathcal{C}$

$\mathbf{A}/\theta$ sera llamada el *algebra cociente de* $\mathbf{A}$ *por* θ.

Lemma 198 *Supongamos* τ *es algebraico. Si* $F : \mathbf{A} \to \mathbf{B}$ *es un homomorfismo, entonces* $\ker F$ *es una congruencia sobre* $\mathbf{A}$

Proof. Sea $f \in \mathcal{F}_n$. Supongamos que $a_1, ..., a_n, b_1, ..., b_n \in A$ son tales que $a_1 \ker F b_1, ..., a_n \ker F b_n$. Tenemos entonces que

$$\begin{array}{rcl} F(f^{\mathbf{A}}(a_1, ..., a_n)) & = & f^{\mathbf{B}}(F(a_1), ..., F(a_n)) \\ & = & f^{\mathbf{B}}(F(b_1), ..., F(b_n)) \\ & = & F(f^{\mathbf{A}}(b_1, ..., b_n)) \end{array}$$

lo cual nos dice que $f^{\mathbf{A}}(a_1, ..., a_n) \ker F f^{\mathbf{A}}(b_1, ..., b_n)$ ■

Al mapeo

$$\begin{array}{rcl} A & \to & A/\theta \\ a & \to & a/\theta \end{array}$$

lo llamaremos la *proyeccion canonica* y lo denotaremos con π_θ.

Lemma 199 $\pi_\theta : \mathbf{A} \to \mathbf{A}/\theta$ *es un homomorfismo cuyo nucleo es* θ

Proof. Sea $c \in \mathcal{C}$. Tenemos que

$$\pi_\theta(c^{\mathbf{A}}) = c^{\mathbf{A}}/\theta = c^{\mathbf{A}/\theta}$$

Sea $f \in \mathcal{F}_n$, con $n \geq 1$ y sean $a_1, ..., a_n \in A$. Tenemos que

$$\begin{array}{rcl} \pi_\theta(f^{\mathbf{A}}(a_1, ..., a_n)) & = & f^{\mathbf{A}}(a_1, ..., a_n)/\theta \\ & = & f^{\mathbf{A}/\theta}(a_1/\theta, ..., a_n/\theta) \\ & = & f^{\mathbf{A}/\theta}(\pi_\theta(a_1), ..., \pi_\theta(a_n)) \end{array}$$

con lo cual π_θ es un homomorfismo. Es trivial que $\ker \pi_\theta = \theta$ ■

Corollary 200 *Para cada* $t \in T^\tau$, $\vec{a} \in A^{\mathbf{N}}$, *se tiene que* $t^{\mathbf{A}/\theta}[(a_1/\theta, a_2/\theta, ...)] = t^{\mathbf{A}}[(a_1, a_2, ...)]/\theta$.

Proof. Ya que π_θ es un homomorfismo, se puede aplicar el Lema 194. ■

Theorem 201 *Sea* $F : \mathbf{A} \to \mathbf{B}$ *un homomorfismo sobreyectivo. Entonces*

$$\begin{array}{lll} A/\ker F & \to & B \\ a/\ker F & \to & F(a) \end{array}$$

define sin ambiguedad una funcion $\bar{F}$ *la cual es un isomorfismo de* $\mathbf{A}/\ker F$ *en* $\mathbf{B}$

Proof. Notese que la definicion de $\bar{F}$ es inambigua ya que si $a/\ker F = a'/\ker F$, entonces $F(a) = F(a')$. Ya que F es sobre, tenemos que $\bar{F}$ lo es. Supongamos que $\bar{F}(a/\ker F) = \bar{F}(a'/\ker F)$. Claramente entonces tenemos que $F(a) = F(a')$, lo cual nos dice que $a/\ker F = a'/\ker F$. Esto prueba que $\bar{F}$ es inyectiva. Para ver que $\bar{F}$ es un isomorfismo, por el Lema 196, basta con ver que $\bar{F}$ es un homomorfismo. Sea $c \in \mathcal{C}$. Tenemos que

$$\bar{F}(c^{\mathbf{A}/\ker F}) = \bar{F}(c^{\mathbf{A}}/\ker F) = F(c^{\mathbf{A}}) = c^{\mathbf{B}}$$

Sea $f \in \mathcal{F}_n$. Sean $a_1, ..., a_n \in A$. Tenemos que

$$\begin{array}{lll} \bar{F}(f^{\mathbf{A}/\ker F}(a_1/\ker F, ..., a_n/\ker F)) & = & \bar{F}(f^{\mathbf{A}}(a_1, ..., a_n)/\ker F) \\ & = & F(f^{\mathbf{A}}(a_1, ..., a_n)) \\ & = & f^{\mathbf{B}}(F(a_1), ..., F(a_n)) \\ & = & f^{\mathbf{B}}(\bar{F}(a_1/\ker F), ..., \bar{F}(a_n/\ker F)) \end{array}$$

con lo cual $\bar{F}$ cunple (2) de la definicion de homomorfismo ■

Producto directo de algebras Dadas τ-algebras $\mathbf{A}, \mathbf{B}$, definamos una nueva τ-algebra $\mathbf{A} \times \mathbf{B}$, de la siguiente manera

(1) Universo de $\mathbf{A} \times \mathbf{B} = A \times B$

(2) $c^{\mathbf{A}\times\mathbf{B}} = (c^{\mathbf{A}}, c^{\mathbf{B}})$, para cada $c \in \mathcal{C}$

(3) $f^{\mathbf{A}\times\mathbf{B}}((a_1, b_1), ..., (a_n, b_n)) = (f^{\mathbf{A}}(a_1, ..., a_n), f^{\mathbf{B}}(b_1, ..., b_n))$, para cada $f \in \mathcal{F}_n$

Llamaremos a $\mathbf{A} \times \mathbf{B}$ el *producto directo* de $\mathbf{A}$ y $\mathbf{B}$.

Los mapeos

$$\begin{array}{lll} \pi_1 : A \times B & \to & A \\ (a, b) & \to & a \end{array} \qquad \begin{array}{lll} \pi_2 : A \times B & \to & B \\ (a, b) & \to & b \end{array}$$

seran llamados las *proyecciones canonicas* asociadas al producto $A \times B$

Lemma 202 *Los mapeos* $\pi_1 : \mathbf{A} \times \mathbf{B} \to \mathbf{A}$ *y* $\pi_2 : \mathbf{A} \times \mathbf{B} \to \mathbf{B}$ *son homomorfismos*

Proof. Veamos que π_1 es un homomorfismo. Primero notese que si $c \in \mathcal{C}$, entonces

$$\pi_1(c^{\mathbf{A}\times\mathbf{B}}) = \pi_1((c^{\mathbf{A}}, c^{\mathbf{B}})) = c^{\mathbf{A}}$$

Sea $f \in \mathcal{F}_n$, con $n \geq 1$ y sean $(a_1, b_1), ..., (a_n, b_n) \in A \times B$. Tenemos que

$$\begin{array}{rcl} \pi_1(f^{\mathbf{A}\times\mathbf{B}}((a_1,b_1),...,(a_n,b_n)) & = & \pi_1((f^{\mathbf{A}}(a_1,...,a_n), f^{\mathbf{B}}(b_1,...,b_n)) \\ & = & f^{\mathbf{A}}(a_1,...,a_n) \\ & = & f^{\mathbf{A}}(\pi_1(a_1,b_1),...,\pi_1(a_n,b_n)) \end{array}$$

con lo cual hemos probado que π_1 cumple (2) de la definicion de homomorfismo
■

Lemma 203 *Para cada* $t \in T^\tau$, $((a_1, b_1), (a_2, b_2), ...) \in (A \times B)^{\mathbf{N}}$, *se tiene que* $t^{\mathbf{A}\times\mathbf{B}}[((a_1,b_1),(a_2,b_2),...)] = (t^{\mathbf{A}}[(a_1,a_2,...)], t^{\mathbf{B}}[(b_1,b_2,...)])$

6.10 Dos teoremas de reemplazo

Probaremos dos teoremas muy importantes que en algun sentido nos dicen que el reemplazo sintactico se lleva bien con la semantica. Antes nos dedicaremos a desarrollar la notacion declaratoria la cual hace mas intuitivos los resultados y ademas permite manejar las cosas con mas naturalidad y practicidad.

6.10.1 Notacion declaratoria para terminos

Si t es un termino de tipo τ, entonces escribiremos $t =_d t(v_1, ..., v_n)$ para declarar que $v_1, ..., v_n$ son variables distintas (con $n \geq 1$) y tales que toda variable que ocurre en t pertenece a $\{v_1, ..., v_n\}$ (no necesariamente toda v_j debe ocurrir en t).

El uso de declaraciones de la forma $t =_d t(v_1, ..., v_n)$ sera muy util cuando se lo convina con ciertas convenciones notacionales que describiremos a continuacion.

Convencion Notacional 1: Cuando hayamos hecho la declaracion $t =_d t(v_1, ..., v_n)$, si $P_1, ..., P_n$ son palabras cualesquiera (no necesariamente terminos), entonces $t(P_1, ..., P_n)$ denotara la palabra que resulta de reemplazar (simultaneamente) cada ocurrencia de v_1 en t, por P_1, cada ocurrencia de v_2 en t, por P_2, etc.

Notese que cuando las palabras $P_i's$ son terminos, $t(P_1, ..., P_n)$ es un termino (Lema 181). Ademas notese que en esta convencion notacional, el orden de las variables $v_1, ..., v_n$ es clave. Por ejemplo si $\tau = (\emptyset, \{\mathrm{FU}\}, \emptyset, \{(\mathrm{FU}, 2)\})$ y $t = \mathrm{FU}(\mathrm{FU}(x_2, x_{16}), x_3)$ y declaramos $t =_d t(x_3, x_2, x_{16})$, entonces

- Si declaramos $t =_d t(x_3, x_2, x_{16})$, entonces $t(\#\#, \blacktriangle\#\blacktriangle, @@)$ denotara la palabra $\text{FU}(\text{FU}(\blacktriangle\#\blacktriangle, @@), \#\#)$

- Si declaramos $t =_d t(x_{16}, x_3, x_2)$, entonces $t(\#\#, \blacktriangle\#\blacktriangle, @@)$ denotara la palabra $\text{FU}(\text{FU}(@@, \#\#), \blacktriangle\#\blacktriangle)$

Tambien podriamos haber declarado $t =_d t(x_3, x_{200}, x_2, x_{16}, x_{100})$ y en tal caso $t(\#\#, !!!!, \blacktriangle\#\blacktriangle, @@, !!)$ denotara la palabra $\text{FU}(\text{FU}(\blacktriangle\#\blacktriangle, @@), \#\#)$

Convencion Notacional 2: Cuando hayamos declarado $t =_d t(v_1, ..., v_n)$, si $\mathbf{A}$ es un modelo de tipo τ y $a_1, ..., a_n \in A$, entonces con $t^{\mathbf{A}}[a_1..., a_n]$ denotaremos al elemento $t^{\mathbf{A}}[\vec{b}]$, donde $\vec{b}$ es una asignacion tal que a cada v_i le asigna el valor a_i. (Notese que esta notacion es inhambigua gracias al Lema 190.)

Nuevamente cabe destacar que en esta convencion notacional, el orden de las variables $v_1, ..., v_n$ es clave. Por ejemplo si τ y t son los dados en el ejemplo anterior y $\mathbf{A}$ es dado por $A = \{1, 2, 3\}$ y $\text{FU}^{\mathbf{A}}(i, j) = j$, para cada $i, j \in A$, tenemos que

- $t^{\mathbf{A}}[2, 1, 3] = 2$ si declaramos $t =_d t(x_3, x_2, x_{16})$
- $t^{\mathbf{A}}[2, 1, 3] = 1$ si declaramos $t =_d t(x_{16}, x_3, x_2)$

Tambien podriamos haber declarado $t =_d t(x_3, x_{200}, x_2, x_{16}, x_{100})$ y en tal caso $t^{\mathbf{A}}[2, 10, 1, 3, 1000] = 2$.

Para establecer nuestra Convencion Notacional 3, debemos antes enunciar un lema de "lectura de terminos una ves declarados" el cual el lector no tendra inconvenientes en probar.

Lemma 204 *Sea τ un tipo cualquiera y supongamos $t \in T^{\tau}$. Si $t =_d t(v_1, ..., v_n)$, entonces se da alguna de las siguientes*

(1) $t = c$, para algun $c \in \mathcal{C}$

(2) $t = v_j$, para algun j

(3) $t = f(t_1, ..., t_m)$, con $f \in \mathcal{F}_m$ y $t_1, ..., t_m \in T^{\tau}$ tales que las variables que ocurren en cada uno de ellos estan en $\{v_1, ..., v_n\}$

Proof. Rutina ■

Convencion Notacional 3: Cuando hayamos declarado $t =_d t(v_1, ..., v_n)$ y se el caso (3) del Lema 204 supondremos tacitamente que tambien hemos hecho las declaraciones $t_1 =_d t_1(v_1, ..., v_n), ..., t_m =_d t_m(v_1, ..., v_n)$.

Cabe destacar que esta ultima convencion notacional junto con la Convencion Notacional 1, nos dice que cuando se de el caso (3) del Lema 204, si $P_1, ..., P_n$ son palabras cualesquiera, entonces

$$t(P_1, ..., P_n) = f(t_1(P_1, ..., P_n), ..., t_m(P_1, ..., P_n))$$

El siguiente lema se basa en la Convencion Notacional 3 y nos permite darle caracter recursivo a la notacion $t^{\mathbf{A}}[a_1, ..., a_n]$.

Lemma 205 *Sea τ un tipo cualquiera y $t \in T^\tau$. Supongamos $t =_d t(v_1, ..., v_n)$. Sea $\mathbf{A}$ un modelo de tipo τ. Sean $a_1, ..., a_n \in A$. Se tiene que:*

(1) Si $t = c$, entonces $t^{\mathbf{A}}[a_1, ..., a_n] = c^{\mathbf{A}}$

(2) Si $t = v_j$, entonces $t^{\mathbf{A}}[a_1, ..., a_n] = a_j$

(3) Si $t = f(t_1, ..., t_m)$, con $f \in \mathcal{F}_m$ y $t_1, ..., t_m \in T^\tau$, entonces

$$t^{\mathbf{A}}[a_1, ..., a_n] = f^{\mathbf{A}}(t_1^{\mathbf{A}}[a_1, ..., a_n], ..., t_m^{\mathbf{A}}[a_1, ..., a_n])$$

Proof. (1) y (2) son triviales.

(3) Sea $\vec{b}$ una asignacion tal que a cada v_i le asigna el valor a_i. Tenemos que

$$\begin{array}{rcl} t^{\mathbf{A}}[a_1, ..., a_n] & = & t^{\mathbf{A}}[\vec{b}] \text{ (por def. de } t^{\mathbf{A}}[a_1, ..., a_n]) \\ & = & f^{\mathbf{A}}(t_1^{\mathbf{A}}[\vec{b}], ..., t_m^{\mathbf{A}}[\vec{b}]) \text{ (por def. de } t^{\mathbf{A}}[\vec{b}]) \\ & = & f^{\mathbf{A}}(t_1^{\mathbf{A}}[a_1, ..., a_n], ..., t_m^{\mathbf{A}}[a_1, ..., a_n]) \text{ (por def. de cada } t_i^{\mathbf{A}}[a_1, ..., a_n]) \end{array}$$

■

6.10.2 Teorema de reemplazo para terminos

Ahora si podemos enunciar y probar el primero de nuestros teoremas de reemplazo

Theorem 206 *Supongamos $t =_d t(w_1, ..., w_k)$, $s_1 =_d s_1(v_1, ..., v_n), ..., s_k =_d s_k(v_1, ..., v_n)$. Todas las variables de $t(s_1, ..., s_k)$ estan en $\{v_1, ..., v_n\}$ y si declaramos $t(s_1, ..., s_k) =_d t(s_1, ..., s_k)(v_1, ..., v_n)$, entonces para cada estructura $\mathbf{A}$ y $a_1, ..., a_n \in A$, se tiene que*

$$t(s_1, ..., s_k)^{\mathbf{A}}[a_1, ..., a_n] = t^{\mathbf{A}}[s_1^{\mathbf{A}}[a_1, ..., a_n], ..., s_k^{\mathbf{A}}[a_1, ..., a_n]].$$

Proof. Por induccion en el l tal que $t \in T_l^\tau$. El caso $l = 0$ es dejado al lector. Supongamos entonces que el teorema vale siempre que $t \in T_l^\tau$ y veamos que entonces vale cuando $t \in T_{l+1}^\tau - T_l^\tau$. Por el Lema 204 hay $f \in \mathcal{F}_m$ y $t_1, ..., t_m$ terminos tales $t = f(t_1, ..., t_m)$ y las variables que ocurren en cada t_i estan en $\{w_1, ..., w_k\}$. Por la unicidad de la lectura de terminos tenemos que

$t_1, ..., t_m \in T_l^\tau$ (por que?). Notese que por nuestra Convencion Notacional 3 asumimos ya hechas las declaraciones

$$t_1 =_d t_1(w_1, ..., w_k), ..., t_m =_d t_m(w_1, ..., w_k)$$

Por HI tenemos que las variables de cada $t_i(s_1, ..., s_k)$ estan en $\{v_1, ..., v_n\}$, lo cual nos permite hacer las siguientes declaraciones:

$$t_i(s_1, ..., s_k) =_d t_i(s_1, ..., s_k)(v_1, ..., v_n),\ i = 1, ..., m$$

Por HI tenemos entonces que

$$t_i(s_1, ..., s_k)^{\mathbf{A}}[\vec{a}] = t_i^{\mathbf{A}}[s_1^{\mathbf{A}}[\vec{a}], ..., s_k^{\mathbf{A}}[\vec{a}]],\ i = 1, ..., m$$

Ya que las variables de cada $t_i(s_1, ..., s_k)$ estan en $\{v_1, ..., v_n\}$, tenemos que las variables de $t(s_1, ..., s_k) = f(t_1(s_1, ..., s_k), ..., t_m(s_1, ..., s_k))$ estan en $\{v_1, ..., v_n\}$. Declaremos entonces $t(s_1, ..., s_k) =_d t(s_1, ..., s_k)(v_1, ..., v_n)$. Solo nos falta probar que

$$t(s_1, ..., s_k)^{\mathbf{A}}[a_1, ..., a_n] = t^{\mathbf{A}}[s_1^{\mathbf{A}}[a_1, ..., a_n], ..., s_k^{\mathbf{A}}[a_1, ..., a_n]].$$

lo cual se detalla a continuacion

$$\begin{array}{rcl} t(s_1, ..., s_k)^{\mathbf{A}}[\vec{a}] & = & f(t_1(s_1, ..., s_k), ..., t_m(s_1, ..., s_k))^{\mathbf{A}}[\vec{a}] \\ & = & f^{\mathbf{A}}(t_1(s_1, ..., s_k)^{\mathbf{A}}[\vec{a}], ..., t_m(s_1, ..., s_k)^{\mathbf{A}}[\vec{a}]) \\ & = & f^{\mathbf{A}}(t_1^{\mathbf{A}}[s_1^{\mathbf{A}}[\vec{a}], ..., s_k^{\mathbf{A}}[\vec{a}]], ..., t_m^{\mathbf{A}}[s_1^{\mathbf{A}}[\vec{a}], ..., s_k^{\mathbf{A}}[\vec{a}]]) \\ & = & t^{\mathbf{A}}[s_1^{\mathbf{A}}[\vec{a}], ..., s_k^{\mathbf{A}}[\vec{a}]] \end{array}$$

■

6.10.3 Notacion declaratoria para formulas

Si φ es una formula de tipo τ, entonces escribiremos $\varphi =_d \varphi(v_1, ..., v_n)$ para declarar que $v_1, ..., v_n$ son variables distintas (con $n \geq 1$) tales que $Li(\varphi) \subseteq \{v_1, ..., v_n\}$. Tal como para el caso de terminos, el uso de declaraciones de la forma $\varphi =_d \varphi(v_1, ..., v_n)$ sera muy util cuando se convina con ciertas convenciones notacionales que describiremos a continuacion.

Convencion Notacional 4: Cuando hayamos hecho la declaracion $\varphi =_d \varphi(v_1, ..., v_n)$, si $P_1, ..., P_n$ son palabras cualesquiera, entonces $\varphi(P_1, ..., P_n)$ denotara la palabra que resulta de reemplazar (simultaneamente) cada ocurrencia libre de v_1 en φ, por P_1, cada ocurrencia libre de v_2 en φ, por P_2, etc.

Notese que cuando las palabras $P_i's$ son terminos, $\varphi(P_1, ..., P_n)$ es una formula. Ademas notese que tal como para el caso de terminos, en esta convencion notacional, el orden de las variables $v_1, ..., v_n$ es clave. Es facil dar el ejemplo analogo al dado para terminos.

Convencion Notacional 5: Cuando hayamos declarado $\varphi =_d \varphi(v_1, ..., v_n)$, si $\mathbf{A}$ es un modelo de tipo τ y $a_1, ..., a_n \in A$, entonces $\mathbf{A} \models \varphi[a_1..., a_n]$ significara que $\mathbf{A} \models \varphi[\vec{b}]$, donde $\vec{b}$ es una asignacion tal que a cada v_i le asigna el valor a_i. (Notese que esta definicion es inambigua gracias al Lema 191). En gral $\mathbf{A} \not\models \varphi[a_1, ..., a_n]$ significara que no sucede $\mathbf{A} \models \varphi[a_1, ..., a_n]$

Nuevamente cabe destacar que en esta convencion notacional, el orden de las variables $v_1, ..., v_n$ es clave y dejamos al lector encontrar un ejemplo donde esto se vea claramente. Para establecer nuestra Convencion Notacional 6, debemos antes enunciar un "lema de lectura unica de formulas declaradas" el cual el lector no tendra inconvenientes en probar.

Lemma 207 *Sea τ un tipo cualquiera y $\varphi \in F^\tau$. Supongamos $\varphi =_d \varphi(v_1, ..., v_n)$. Entonces se una y solo una de las siguientes:*

(1) $\varphi = (t \equiv s)$, con $t, s \in T^\tau$, unicos y tales que las variables que ocurren en t o en s estan todas en $\{v_1, ..., v_n\}$

(2) $\varphi = r(t_1, ..., t_m)$, con $r \in \mathcal{R}_m$ y $t_1, ..., t_m \in T^\tau$, unicos y tales que las variables que ocurren en cada t_i estan todas en $\{v_1, ..., v_n\}$

(3) $\varphi = (\varphi_1 \wedge \varphi_2)$, con $\varphi_1, \varphi_2 \in F^\tau$, unicas y tales que $Li(\varphi_1) \cup Li(\varphi_2) \subseteq \{v_1, ..., v_n\}$

(4) $\varphi = (\varphi_1 \vee \varphi_2)$, con $\varphi_1, \varphi_2 \in F^\tau$, unicas y tales que $Li(\varphi_1) \cup Li(\varphi_2) \subseteq \{v_1, ..., v_n\}$

(5) $\varphi = (\varphi_1 \rightarrow \varphi_2)$, con $\varphi_1, \varphi_2 \in F^\tau$, unicas y tales que $Li(\varphi_1) \cup Li(\varphi_2) \subseteq \{v_1, ..., v_n\}$

(6) $\varphi = (\varphi_1 \leftrightarrow \varphi_2)$, con $\varphi_1, \varphi_2 \in F^\tau$, unicas y tales que $Li(\varphi_1) \cup Li(\varphi_2) \subseteq \{v_1, ..., v_n\}$

(7) $\varphi = \neg\varphi_1$, con $\varphi_1 \in F^\tau$, unica y tal que $Li(\varphi_1) \subseteq \{v_1, ..., v_n\}$

(8) $\varphi = \forall v_j \varphi_1$, con $v_j \in \{v_1, ..., v_n\}$ y $\varphi_1 \in F^\tau$, unicas y tales que $Li(\varphi_1) \subseteq \{v_1, ..., v_n\}$

(9) $\varphi = \forall v \varphi_1$, con $v \in Var - \{v_1, ..., v_n\}$ y $\varphi_1 \in F^\tau$, unicas y tales que $Li(\varphi_1) \subseteq \{v_1, ..., v_n, v\}$

(10) $\varphi = \exists v_j \varphi_1$, con $v_j \in \{v_1, ..., v_n\}$ y $\varphi_1 \in F^\tau$, unicas y tales que $Li(\varphi_1) \subseteq \{v_1, ..., v_n\}$

(11) $\varphi = \exists v \varphi_1$, con $v \in Var - \{v_1, ..., v_n\}$ y $\varphi_1 \in F^\tau$, unicas y tales que $Li(\varphi_1) \subseteq \{v_1, ..., v_n, v\}$

Proof. Induccion en el k tal que $\varphi \in F_k^\tau$ ■

Convencion Notacional 6: Cuando hayamos declarado $\varphi =_d \varphi(v_1, ..., v_n)$, entonces:

- si se da el caso (1) del Lema 207, supondremos tacitamente que tambien hemos hecho las declaraciones $t =_d t(v_1, ..., v_n)$ y $s =_d s(v_1, ..., v_n)$.

- si se da el caso (2) del Lema 207, supondremos tacitamente que tambien hemos hecho las declaraciones $t_1 =_d t_1(v_1, ..., v_n), ..., t_m =_d t_m(v_1, ..., v_n)$.

- si se da cualquiera de los casos (3), (4), (5) o (6) del Lema 207, supondremos tacitamente que tambien hemos hecho las declaraciones $\varphi_1 =_d \varphi_1(v_1, ..., v_n)$ y $\varphi_2 =_d \varphi_2(v_1, ..., v_n)$.

- si se da cualquiera de los casos (7), (8) o (10) del Lema 207, supondremos tacitamente que tambien hemos hecho la declaracion $\varphi_1 =_d \varphi_1(v_1, ..., v_n)$.

- si se da el caso (9) o el caso (11) del Lema 207, supondremos tacitamente que tambien hemos hecho la declaracion $\varphi_1 =_d \varphi_1(v_1, ..., v_n, v)$.

El siguiente lema se basa en la Convencion Notacional 6 y nos permite darle caracter recursivo a la notacion $\mathbf{A} \models \varphi[a_1, ..., a_n]$.

Lemma 208 *Supongamos* $\varphi =_d \varphi(v_1, ..., v_n)$. *Sea* $\mathbf{A} = (A, i)$ *un modelo de tipo* τ *y sean* $a_1, ..., a_n \in A$. *Entonces*

(1) Si $\varphi = (t \equiv s)$, *entonces*

- $\mathbf{A} \models \varphi[a_1, ..., a_n]$ *si y solo si* $t^{\mathbf{A}}[a_1, ..., a_n] = s^{\mathbf{A}}[a_1, ..., a_n]$

(2) Si $\varphi = r(t_1, ..., t_m)$, *entonces*

- $\mathbf{A} \models \varphi[a_1, ..., a_n]$ *si y solo si* $(t_1^{\mathbf{A}}[a_1, ..., a_n], ..., t_m^{\mathbf{A}}[a_1, ..., a_n]) \in r^{\mathbf{A}}$

(3) Si $\varphi = (\varphi_1 \wedge \varphi_2)$, *entonces*

- $\mathbf{A} \models \varphi[a_1, ..., a_n]$ *si y solo si* $\mathbf{A} \models \varphi_1[a_1, ..., a_n]$ *y* $\mathbf{A} \models \varphi_2[a_1, ..., a_n]$

(4) Si $\varphi = (\varphi_1 \vee \varphi_2)$, *entonces*

- $\mathbf{A} \models \varphi[a_1, ..., a_n]$ *si y solo si* $\mathbf{A} \models \varphi_1[a_1, ..., a_n]$ *o* $\mathbf{A} \models \varphi_2[a_1, ..., a_n]$

(5) Si $\varphi = (\varphi_1 \rightarrow \varphi_2)$, *entonces*

- $\mathbf{A} \models \varphi[a_1, ..., a_n]$ *si y solo si* $\mathbf{A} \models \varphi_2[a_1, ..., a_n]$ *o* $\mathbf{A} \not\models \varphi_1[a_1, ..., a_n]$

(6) Si $\varphi = (\varphi_1 \leftrightarrow \varphi_2)$, *entonces*

- $\mathbf{A} \models \varphi[a_1, ..., a_n]$ *si y solo si ya sea* $\mathbf{A} \models \varphi_1[a_1, ..., a_n]$ *y* $\mathbf{A} \models \varphi_2[a_1, ..., a_n]$ *o* $\mathbf{A} \not\models \varphi_1[a_1, ..., a_n]$ *y* $\mathbf{A} \not\models \varphi_2[a_1, ..., a_n]$

(7) Si $\varphi = \neg\varphi_1$, entonces

- $\mathbf{A} \models \varphi[a_1, ..., a_n]$ *si y solo si* $\mathbf{A} \not\models \varphi_1[a_1, ..., a_n]$

(8) Si $\varphi = \forall v_j \varphi_1$, entonces

- $\mathbf{A} \models \varphi[a_1, ..., a_n]$ *si y solo si* $\mathbf{A} \models \varphi_1[a_1, ..., a, ..., a_n]$*, para todo* $a \in A$.

(9) Si $\varphi = \forall v \varphi_1$, con $v \notin \{v_1, ..., v_n\}$, entonces

- $\mathbf{A} \models \varphi[a_1, ..., a_n]$ *si y solo si* $\mathbf{A} \models \varphi_1[a_1, ..., a_n, a]$*, para todo* $a \in A$.

(10) Si $\varphi = \exists v_j \varphi_1$, entonces

- $\mathbf{A} \models \varphi[a_1, ..., a_n]$ *si y solo si* $\mathbf{A} \models \varphi_1[a_1, ..., a, ..., a_n]$*, para algun* $a \in A$.

(11) Si $\varphi = \exists v \varphi_1$, con $v \notin \{v_1, ..., v_n\}$, entonces

- $\mathbf{A} \models \varphi[a_1, ..., a_n]$ *si y solo si* $\mathbf{A} \models \varphi_1[a_1, ..., a_n, a]$*, para algun* $a \in A$.

Proof. Rutina. ■

6.10.4 Alcance de la ocurrencia de un cuantificador en una formula

Lemma 209 *Si Qv ocurre en φ a partir de i, entonces hay una unica formula ψ tal que $Qv\psi$ ocurre en φ a partir de i.*

Proof. Por induccion en el k tal que $\varphi \in F^\tau$. ■

Dada una ocurrencia de Qv en una formula φ, la formula ψ del lema anterior sera llamada el *alcance* de dicha ocurrencia en φ. Notese que dos ocurrencias distintas de Qv en φ pueden tener alcances distintos.

6.10.5 Sustitucion de variables libres

Diremos que *v es sustituible por w en φ* cuando ninguna ocurrencia libre de v en φ sucede dentro de una ocurrencia de una subformula de la forma $Qw\psi$ en φ. En otras palabras v no sera sustituible por w en φ cuando alguna ocurrencia libre de v en φ suceda dentro de una ocurrencia en φ de una formula de la forma $Qw\psi$. Notese que puede suceder que v sea sustituible por w en φ y que sin envargo haya una subformula de la forma $Qw\psi$ para la cual $v \in Li(Qw\psi)$. Dejamos como ejercicio encontrar un ejemplo de esta situacion.

Usando lemas anteriores podemos ver que se dan las siguientes propiedades

(1) Si φ es atomica, entonces v es sustituible por w en φ

(2) Si $\varphi = (\varphi_1 \eta \varphi_2)$, entonces v es sustituible por w en φ sii v es substituible por w en φ_1 y v es

substituible por w en φ_2

(3) Si $\varphi = \neg\varphi_1$, entonces v es sustituible por w en φ sii v es substituible por w en φ_1

(4) Si $\varphi = Qv\varphi_1$, entonces v es sustituible por w en φ

(5) Si $\varphi = Qw\varphi_1$ y $v \in Li(\varphi_1)$, entonces v no es sustituible por w en φ

(6) Si $\varphi = Qw\varphi_1$ y $v \notin Li(\varphi_1)$, entonces v es sustituible por w en φ

(7) Si $\varphi = Qu\varphi_1$, con $u \neq v, w$, entonces v es sustituible por w en φ sii v es sustituible por w en φ_1

Notese que las propiedades (1),...,(7) pueden usarse para dar una definicion recursiva de la relacion "*v es sustituible por w en φ*".

6.10.6 Teorema de reemplazo para formulas

Ahora si podemos enunciar y probar el primero de nuestros teoremas de reemplazo. Antes una definicion. Dado un termino t, diremos que una variable *v es sustituible por t en φ* cuando v sea sustituible en φ por cada variable que ocurre en t.

Theorem 210 *Supongamos $\varphi =_d \varphi(w_1, ..., w_k)$, $t_1 =_d t_1(v_1, ..., v_n), ..., t_k =_d t_k(v_1, ..., v_n)$ y supongamos ademas que cada w_j es sustituible por t_j en φ. Entonces*

(a) $Li(\varphi(t_1, ..., t_k)) \subseteq \{v_1, ..., v_n\}$

(b) Si declaramos $\varphi(t_1, ..., t_k) =_d \varphi(t_1, ..., t_k)(v_1, ..., v_n)$, entonces para cada estructura $\mathbf{A}$ y $\vec{a} \in A^n$ se tiene

$$\mathbf{A} \models \varphi(t_1, ..., t_k)[\vec{a}] \text{ si y solo si } \mathbf{A} \models \varphi[t_1^{\mathbf{A}}[\vec{a}], ..., t_k^{\mathbf{A}}[\vec{a}]]$$

Proof. Probaremos que se dan (a) y (b), por induccion en el l tal que $\varphi \in F_l^\tau$. El caso $l = 0$ es una consecuencia directa del Teorema 206. Supongamos (a) y (b) valen para cada $\varphi \in F_l^\tau$ y sea $\varphi \in F_{l+1}^\tau - F_l^\tau$. Notese que se puede suponer que cada v_i ocurre en algun t_i, y que cada $w_i \in Li(\varphi)$, ya que para cada φ, el caso general se desprende del caso con estas restricciones. Hay varios casos

CASO $\varphi = \forall w \varphi_1$, con $w \notin \{w_1, ..., w_k\}$.

Notese que cada $w_j \in Li(\varphi_1)$. Ademas notese que $w \notin \{v_1, ..., v_n\}$ ya que de lo contrario w ocurriria en algun t_j, y entonces w_j no seria sustituible por t_j en φ. Sean

$$\begin{array}{rcl} \tilde{t}_1 & = & t_1 \\ & \vdots & \\ \tilde{t}_k & = & t_k \\ \tilde{t}_{k+1} & = & w \end{array}$$

Declaremos
$$\tilde{t}_j =_d \tilde{t}_j(v_1, ..., v_n, w)$$

Notese que nuestra Convencion Notacional 6 nos dice que tenemos implicitamente hecha la declaracion $\varphi_1 =_d \varphi_1(w_1, ..., w_k, w)$. Por (a) de la hipotesis inductiva tenemos que

$$Li(\varphi_1(t_1, ..., t_k, w)) = Li(\varphi_1(\tilde{t}_1, ..., \tilde{t}_k, \tilde{t}_{k+1})) \subseteq \{v_1, ..., v_n, w\}$$

y por lo tanto
$$Li(\varphi(t_1, ..., t_k)) \subseteq \{v_1, ..., v_n\}$$

lo cual prueba (a). Finalmente para probar (b) declaremos $\varphi(t_1, ..., t_k) =_d \varphi(t_1, ..., t_k)(v_1, ..., v_n)$. Se tiene que

$$\begin{array}{c}
\mathbf{A} \models \varphi(t_1, ..., t_k)[\vec{a}] \\
\Updownarrow \\
\mathbf{A} \models \varphi_1(\tilde{t}_1, ..., \tilde{t}_k, \tilde{t}_{k+1})[\vec{a}, a], \text{ para todo } a \in A \\
\Updownarrow \text{ (por HI)} \\
\mathbf{A} \models \varphi_1[\tilde{t}_1^{\mathbf{A}}[\vec{a}, a], ..., \tilde{t}_k^{\mathbf{A}}[\vec{a}, a], \tilde{t}_{k+1}^{\mathbf{A}}[\vec{a}, a]], \text{ para todo } a \in A \\
\Updownarrow \\
\mathbf{A} \models \varphi_1[t_1^{\mathbf{A}}[\vec{a}], ..., t_k^{\mathbf{A}}[\vec{a}], a], \text{ para todo } a \in A \\
\Updownarrow \\
\mathbf{A} \models \varphi[t_1^{\mathbf{A}}[\vec{a}], ..., t_k^{\mathbf{A}}[\vec{a}]]
\end{array}$$

lo cual pueba (b). Dejamos al lector los casos restantes. ■

Ejemplo: Sea $\tau = (\emptyset, \{f\}, \emptyset, \{(f, 1)\})$. Sean $\varphi = \exists v_1(f(v_1) \equiv w_1)$ y $t = v_1$, donde v_1 y w_1 son variables distintas. Declaremos $\varphi =_d \varphi(w_1)$ y $t =_d t(v_1)$. Notese que w_1 no es sustituible en φ por t, por lo cual el teorema anterior no se puede aplicar. De hecho la conclusion del teorema no se da en este caso ya que puede verse facilmente que, cualesquiera sea la estructura de tipo τ, $\mathbf{A}$ y $a_1 \in A$, tenemos que:

(a) $\mathbf{A} \models \varphi(t)[a_1]$ si y solo si $f^{\mathbf{A}}$ tiene un pto fijo, es decir, $f^{\mathbf{A}}(a) = a$, para algun $a \in A$

(b) $\mathbf{A} \models \varphi[t^{\mathbf{A}}[a_1]]$ si y solo si a_1 esta en la imagen de $f^{\mathbf{A}}$

las cuales son condiciones claramente no equivalentes.

6.11 Teorias de primer orden

En esta seccion nos avocaremos a dar una solucion al punto (3) de nuestro programa de logica dado en la Seccion 6.6. Una *teoria de primer orden* sera un par (Σ, τ), donde τ es un tipo y Σ es un conjunto de sentencias de tipo τ.

Los elementos de Σ seran llamados *axiomas propios* de (Σ, τ). Un *modelo de* (Σ, τ) sera una estructura de tipo τ la cual satisfaga todos los axiomas propios de (Σ, τ).

Obviamente el concepto de teoria de primer orden es un modelo matematico de nuestro concepto de teoria elemental, el cual es un concepto informal o intuitivo ya que esta basado en el concepto de sentencia elemental.

Algunos ejemplos de teorias de primer orden:

La teoria *Po*. Sea

$$Po = (\{\mathrm{A}_{\leq R}, \mathrm{A}_{\leq T}, \mathrm{A}_{\leq A}\}, \tau_{Po})$$

donde τ_{Po} es el tipo de los posets, es decir $(\emptyset, \emptyset, \{\leq\}, \{(\leq, 2)\})$ y

$$\begin{array}{lcl} \mathrm{A}_{\leq R} & = & \forall x_1 \leq(x_1, x_1) \\ \mathrm{A}_{\leq T} & = & \forall x_1 \forall x_2 \forall x_3 \ ((\leq(x_1, x_2) \wedge \leq(x_2, x_3)) \rightarrow \leq(x_1, x_3)) \\ \mathrm{A}_{\leq A} & = & \forall x_1 \forall x_2 \ ((\leq(x_1, x_2) \wedge \leq(x_2, x_1)) \rightarrow (x_1 \equiv x_2)) \end{array}$$

Notese que una estructura $\mathbf{A}$ de tipo τ_{Po} es un modelo de Po si y solo si $\leq^{\mathbf{A}}$ es un orden parcial sobre A. Estrictamente hablando un modelo de Po no es un poset ya que es un par (A, i) donde A es un conjunto no vacio e i es una funcion con dominio $\{\leq\}$ tal que $i(\leq)$ es un orden parcial sobre A. Es decir, un modelo de Po es un par $(A, \{(\leq, R)\})$ donde A es un conjunto no vacio y R es un orden parcial sobre A. De todas maneras deberia quedar claro que en esencia un poset y un modelo de Po son la misma cosa por lo cual llamaremos a Po la *teoria de los posets* y muchas veces nos referiremos a los modelos de Po como si fueran posets. Dejamos al lector el ejercicio de encontrar una biyeccion natural entre la clase de los modelos de Po y la clase de los posets.

La teoria *RetCua*. Sea $\tau_{RetCua} = (\emptyset, \{\mathsf{s}^2, \mathsf{i}^2\}, \{\leq^2\}, a)$ y sea Σ_{RetCua} el siguiente conjunto de sentencias:

$$\begin{array}{lcl} \mathrm{A}_{\leq R} & = & \forall x_1 \leq(x_1, x_1) \\ \mathrm{A}_{\leq T} & = & \forall x_1 \forall x_2 \forall x_3 \ ((\leq(x_1, x_2) \wedge \leq(x_2, x_3)) \rightarrow \leq(x_1, x_3)) \\ \mathrm{A}_{\leq A} & = & \forall x_1 \forall x_2 \ ((\leq(x_1, x_2) \wedge \leq(x_2, x_1)) \rightarrow (x_1 \equiv x_2)) \\ \mathrm{A}_{\mathsf{s}esC} & = & \forall x_1 \forall x_2 \ (\leq(x_1, \mathsf{s}(x_1, x_2)) \wedge \leq(x_2, \mathsf{s}(x_1, x_2))) \\ \mathrm{A}_{\mathsf{s} \leq C} & = & \forall x_1 \forall x_2 \forall x_3 \ ((\leq(x_1, x_3) \wedge \leq(x_2, x_3)) \rightarrow \leq(\mathsf{s}(x_1, x_2), x_3)) \\ \mathrm{A}_{\mathsf{i}esC} & = & \forall x_1 \forall x_2 \ (\leq(\mathsf{i}(x_1, x_2), x_1) \wedge \leq(\mathsf{i}(x_1, x_2), x_2)) \\ \mathrm{A}_{\mathsf{i} \geq C} & = & \forall x_1 \forall x_2 \forall x_3 \ ((\leq(x_3, x_1) \wedge \leq(x_3, x_2)) \rightarrow \leq(x_3, \mathsf{i}(x_1, x_2))) \end{array}$$

Definamos $RetCua = (\Sigma_{RetCua}, \tau_{RetCua})$. Como pueden notarse los axiomas de $RetCua$ son justamente los axiomas con los que definimos el concepto de reticulado cuaterna en la Seccion 5. Es decir una estructura $\mathbf{A}$ de tipo τ_{RetCua} es modelo de $RetCua$ sii $(A, \mathsf{s}^{\mathbf{A}}, \mathsf{i}^{\mathbf{A}}, \leq^{\mathbf{A}})$ es un reticulado cuaterna. Vayamos mas despacio:

- Una estructura $\mathbf{A}$ de tipo τ_{RetCua} satisface los 3 primeros axiomas de $RetCua$ si y solo si el par $(A, \leq^{\mathbf{A}})$ es un poset. Aqui es muy importante notar que una estructura $\mathbf{A}$ de tipo τ_{RetCua} puede satisfacer los 3 axiomas mencionados antes pero esto no implica que las operaciones $\mathsf{s}^{\mathbf{A}}$ e $\mathsf{i}^{\mathbf{A}}$ deban ser las operaciones infimo y supremo respecto al orden $\leq^{\mathbf{A}}$. De hecho el poset $(A, \leq^{\mathbf{A}})$ podria no tener supremo para algun subconjunto $\{a, b\}$.

- Una estructura $\mathbf{A}$ que satisfaga los 3 primeros axiomas satisfacera el axioma $\mathrm{A}_{\mathsf{s}esC}$ si y solo si cualesquiera sean los elementos $a, b \in A$, se tiene que $a\ \mathsf{s}^{\mathbf{A}}\ b$ es cota superior del conjunto $\{a, b\}$ en el poset $(A, \leq^{\mathbf{A}})$. Por supuesto esto no garaniza que $a\ \mathsf{s}^{\mathbf{A}}\ b$ sea el supremo de $\{a, b\}$ en $(A, \leq^{\mathbf{A}})$, solo nos dice que debe ser una cota superior

- Una estructura $\mathbf{A}$ que satisfaga los 3 primeros axiomas cumplirá $\mathrm{A}_{\mathsf{s}\leq C}$ si y solo si cualesquiera sean los elementos $a, b \in A$, se tiene que $a\ \mathsf{s}^{\mathbf{A}}\ b$ es menor o igual que toda cota superior de $\{a, b\}$ en $(A, \leq^{\mathbf{A}})$. Por supuesto esto tampoco garaniza que $a\ \mathsf{s}^{\mathbf{A}}\ b$ sea el supremo de $\{a, b\}$ en $(A, \leq^{\mathbf{A}})$

De las observaciones anteriores el lector ya se habra dado cuenta que dada una estructura $\mathbf{A}$ de tipo τ_{RetCua} son equivalentes

1. $\mathbf{A}$ es modelo de $RetCua$
2. $(A, \leq^{\mathbf{A}})$ es un poset y cualesquiera sean $a, b \in A$
   (a) $a\ \mathsf{s}^{\mathbf{A}}\ b =$ supremo de $\{a, b\}$ en $(A, \leq^{\mathbf{A}})$
   (b) $a\ \mathsf{i}^{\mathbf{A}}\ b =$ infimo de $\{a, b\}$ en $(A, \leq^{\mathbf{A}})$

6.11.1 Definicion del concepto de prueba formal

Recomendamos al lector repasar el concepto de prueba elemental en una teoria elemental, dado en la Seccion 5. Aqui daremos un modelo matematico del concepto de prueba elemental en una teoria (Σ, τ). Tal como lo hemos visto en numerosos ejemplos, una prueba es una sucecion de sentencias junto con una sucesion de "justificaciones" las cuales van explicando o justificando por que es licito que cada una de dichas sentencias aparezca en la sucesion. Por supuesto nuestra definicion sera precisa y matematica por lo que deberemos trabajar bastante para poder escribirla correctamente. Como objeto matematico una prueba resultara ser un par ordenado de palabras cuya primera coordenada codificara en forma natural la sucesion de sentencias y su segunda coordenada codificara la sucesion de justificaciones.

La formalizacion matematica del concepto de prueba elemental es uno de los grandes logros de la ciencia moderna y este hecho se debe en gran medida a que si elejimos bien la teoria, las pruebas elementales no son ni mas ni menos que las pruebas de la matematica misma por lo cual se tiene una definicion matematica de la deduccion matematica!

Reglas Definiremos una serie de conjuntos los cuales poseen informacion deductiva basica, es decir representan las reglas usuales con las que los matematicos dan pasos dentro de una demostracion (aunque muchas veces ellos lo hacen sin avisar debido a la obviedad de dichas reglas).

Recordemos que si τ es un tipo cualquiera, un termino $t \in T^\tau$ es llamado *cerrado* si ninguna variable es subtermino de t. Con T_c^τ denotamos el conjunto formado por todos los terminos cerrados.

Sean

$$\begin{array}{rcl}
Partic^\tau & = & \{(\forall v\varphi(v), \varphi(t)) : \varphi =_d \varphi(v) \in F^\tau \text{ y } t \in T_c^\tau\} \\
Exist^\tau & = & \{(\varphi(t), \exists v\varphi(v)) : \varphi =_d \varphi(v) \in F^\tau \text{ y } t \in T_c^\tau\} \\
Evoc^\tau & = & \{(\varphi, \varphi) : \varphi \in S^\tau\} \\
Absur^\tau & = & \{((\neg\varphi \to (\psi \wedge \neg\psi)), \varphi) : \varphi, \psi \in S^\tau\} \cup \{((\varphi \to (\psi \wedge \neg\psi)), \neg\varphi) : \varphi, \psi \in S^\tau\} \\
ConjElim^\tau & = & \{((\varphi \wedge \psi), \varphi) : \varphi, \psi \in S^\tau\} \cup \{((\varphi \wedge \psi), \psi) : \varphi, \psi \in S^\tau\} \\
EquivElim^\tau & = & \{((\varphi \leftrightarrow \psi), (\varphi \to \psi)) : \varphi, \psi \in S^\tau\} \cup \{((\varphi \leftrightarrow \psi), (\psi \to \varphi)) : \varphi, \psi \in S^\tau\} \\
DisjInt^\tau & = & \{(\varphi, (\varphi \vee \psi)) : \varphi, \psi \in S^\tau\} \cup \{(\psi, (\varphi \vee \psi)) : \varphi, \psi \in S^\tau\}
\end{array}$$

Diremos que φ *se deduce de* ψ *por la regla de particularizacion* (resp. *existencia, evocacion, absurdo, conjuncion-eliminacion, equivalencia-eliminacion, disjuncion-introduccion*), *con respecto a* τ para expresar que $(\psi, \varphi) \in Partic^\tau$ (resp. $(\psi, \varphi) \in Exist^\tau$, $(\psi, \varphi) \in Evoc^\tau$, $(\psi, \varphi) \in Absur^\tau$, $(\psi, \varphi) \in ConjElim^\tau$, $(\psi, \varphi) \in EquivElim^\tau$, $(\psi, \varphi) \in DisjInt^\tau$).

Sea

$$Commut^\tau = Commut1^\tau \cup Commut2^\tau$$

donde

$$\begin{array}{rcl}
Commut1^\tau & = & \{((t \equiv s), (s \equiv t)) : s, t \in T_c^\tau\} \\
Commut2^\tau & = & \{((\varphi \leftrightarrow \psi), (\psi \leftrightarrow \varphi)) : \varphi, \psi \in S^\tau\}
\end{array}$$

Diremos que φ *se deduce de* ψ *por la regla de commutatividad, con respecto a* τ para expresar que $(\psi, \varphi) \in Commut^\tau$.

Sean

$$\begin{array}{rcl}
ModPon^\tau & = & \{(\varphi, (\varphi \to \psi), \psi) : \varphi, \psi \in S^\tau\} \\
ConjInt^\tau & = & \{(\varphi, \psi, (\varphi \wedge \psi)) : \varphi, \psi \in S^\tau\} \\
EquivInt^\tau & = & \{((\varphi \to \psi), (\psi \to \varphi), (\varphi \leftrightarrow \psi)) : \varphi, \psi \in S^\tau\} \\
DisjElim^\tau & = & \{(\neg\varphi, (\varphi \vee \psi), \psi) : \varphi, \psi \in S^\tau\} \cup \{(\neg\psi, (\varphi \vee \psi), \varphi) : \varphi, \psi \in S^\tau\}
\end{array}$$

Diremos que φ *se deduce de* ψ_1 y ψ_2 *por la regla de Modus Ponens* (resp. *conjuncion-introduccion, equivalencia-introduccion, disjuncion-eliminacion*), *con respecto a* τ para expresar que $(\psi_1, \psi_2, \varphi) \in ModPon^\tau$ (resp. $(\psi_1, \psi_2, \varphi) \in ConjInt^\tau$, $(\psi_1, \psi_2, \varphi) \in EquivInt^\tau$, $(\psi_1, \psi_2, \varphi) \in DisjElim^\tau$). Sea

$$DivPorCas^\tau = \{((\varphi_1 \vee \varphi_2), (\varphi_1 \to \psi), (\varphi_2 \to \psi), \psi) : \varphi_1, \varphi_2, \psi \in S^\tau\}$$

Diremos que φ *se deduce de* ψ_1, ψ_2 y ψ_3 *por la regla de division por casos, con respecto a* τ para expresar que $(\psi_1, \psi_2, \psi_3, \varphi) \in DivPorCas^\tau$. Sea

$$Reemp^\tau = Reemp1^\tau \cup Reemp2^\tau$$

donde

$Reemp1^\tau = \{((t \equiv s), \gamma, \tilde{\gamma}) : s, t \in T_c^\tau,$
$\gamma \in S^\tau$ y $\tilde{\gamma} =$ resultado de reemplazar en γ una ocurrencia de t por $s\}$

$Reemp2^\tau = \{(\forall v_1 ... \forall v_n(\varphi \leftrightarrow \psi), \gamma, \tilde{\gamma}) : \varphi, \psi \in F^\tau,\ Li(\varphi) = Li(\psi) = \{v_1, ..., v_n\},$
$n \geq 0, \gamma \in S^\tau$ y $\tilde{\gamma} =$ resultado de reemplazar en γ una ocurrencia de φ por $\psi\}$

Diremos que φ *se deduce de* ψ_1y ψ_2 *por la regla de reemplazo, con respecto a* τ, para expresar que $(\psi_1, \psi_2, \varphi) \in Reemp^\tau$. Sea

$$Trans^\tau = Trans1^\tau \cup Trans2^\tau \cup Trans3^\tau$$

donde

$$\begin{array}{rcl} Trans1^\tau & = & \{((t \equiv s), (s \equiv u), (t \equiv u)) : t, s, u \in T_c^\tau\} \\ Trans2^\tau & = & \{((\varphi \rightarrow \psi), (\psi \rightarrow \Phi), (\varphi \rightarrow \Phi)) : \varphi, \psi, \Phi \in S^\tau\} \\ Trans3^\tau & = & \{((\varphi \leftrightarrow \psi), (\psi \leftrightarrow \Phi), (\varphi \leftrightarrow \Phi)) : \varphi, \psi, \Phi \in S^\tau\} \end{array}$$

Diremos que φ *se deduce de* ψ_1y ψ_2 *por la regla de transitividad, con respecto a* τ para expresar que $(\psi_1, \psi_2, \varphi) \in Trans^\tau$. Sea

$Generaliz^\tau = \{(\varphi(c), \forall v \varphi(v)) : \varphi =_d \varphi(v) \in F^\tau,\ Li(\varphi) = \{v\}$ y $c \in \mathcal{C}$ no ocurre en $\varphi\}$

Es importante el siguiente

Lemma 211 *Si* $(\varphi_1, \varphi_2) \in Generaliz^\tau$*, entonces el nombre de constante* c *del cual habla la definicion de* $Generaliz^\tau$ *esta univocamente determinado por el par* (φ_1, φ_2).

Proof. Notese que c es el unico nombre de constante que ocurre en φ_1 y no ocurre en φ_2 ■

Escribiremos $(\varphi_1, \varphi_2) \in Generaliz^\tau$ *via* c para expresar que $(\varphi_1, \varphi_2) \in Generaliz^\tau$ y que c es el unico nombre de constante que ocurre en φ_1 y no ocurre en φ_2. Diremos que φ_2 *se deduce de* φ_1 *por la regla de generalizacion con nombre de constante* c*, con respecto a* τ, para expresar que $(\varphi_1, \varphi_2) \in Generaliz^\tau$ *via* c

Sea

$Elec^\tau = \{(\exists v \varphi(v), \varphi(e)) : \varphi =_d \varphi(v) \in F^\tau,\ Li(\varphi) = \{v\}$ y $e \in \mathcal{C}$ no ocurre en $\varphi\}$

Es importante el siguiente

Lemma 212 *Si* $(\varphi_1, \varphi_2) \in Elec^\tau$*, entonces el nombre de constante* e *del cual habla la definicion de* $Elec^\tau$ *esta univocamente determinado por el par* (φ_1, φ_2).

Proof. Notese que e es el unico nombre de constante que ocurre en φ_2 y no ocurre en φ_1. ■

Escribiremos $(\varphi_1, \varphi_2) \in Elec^\tau$ *via* e para expresar que $(\varphi_1, \varphi_2) \in Elec^\tau$ y que e es el unico nombre de constante que ocurre en φ_2 y no ocurre en φ_1. Diremos que φ_2 *se deduce de* φ_1 *por la regla de eleccion con nombre de constante* e, *con respecto a* τ para expresar que $(\varphi_1, \varphi_2) \in Elec^\tau$ *via* e.

Lemma 213 *Sea τ un tipo. Todas las reglas exepto las reglas de eleccion y generalizacion son universales en el sentido que si φ se deduce de $\psi_1, ..., \psi_k$ por alguna de estas reglas, entonces $((\psi_1 \wedge ... \wedge \psi_k) \rightarrow \varphi)$ es una sentencia universalmente valida.*

Proof. Veamos que la regla de existencia es universal. Por definicion, un par de $Exist^\tau$ es siempre de la forma $(\varphi(t), \exists v \varphi(v))$, con $\varphi =_d \varphi(v)$ y $t \in T_c^\tau$. Sea $\mathbf{A}$ una estructura de tipo τ tal que $\mathbf{A} \models \varphi(t)$. Sea $t^{\mathbf{A}}$ el valor que toma t en $\mathbf{A}$. Por el Lema 210 tenemos que $\mathbf{A} \models \varphi\left[t^{\mathbf{A}}\right]$, por lo cual tenemos que $\mathbf{A} \models \exists v \varphi(v)$.

Veamos que la regla de reemplazo es universal. Debemos probar que si $(\psi_1, \psi_2, \varphi) \in Reemp^\tau = Reemp1^\tau \cup Reemp2^\tau$, entonces $((\psi_1 \wedge \psi_2) \rightarrow \varphi)$ es una sentencia universalmente valida. El caso en el que $(\psi_1, \psi_2, \varphi) \in Reemp1^\tau$ es facil y lo dejaremos al lector. Para el caso en el que $(\psi_1, \psi_2, \varphi) \in Reemp2^\tau$ nos hara falta un resultado un poco mas general. Veamos por induccion en k que si se dan las siguientes condiciones

- $\alpha \in F_k^\tau$ y $\varphi, \psi \in F^\tau$
- $\mathbf{A}$ es una estructura de tipo τ
- $\overline{\alpha} =$ resultado de reemplazar en α una ocurrencia de φ por ψ,
- $\mathbf{A} \models \varphi[\vec{a}]$ si y solo si $\mathbf{A} \models \psi[\vec{a}]$, para cada $\vec{a} \in A^{\mathbf{N}}$

entonces se da que

- $\mathbf{A} \models \alpha[\vec{a}]$ si y solo si $\mathbf{A} \models \overline{\alpha}[\vec{a}]$, para cada $\vec{a} \in A^{\mathbf{N}}$.

CASO $k = 0$.
Entonces α es atomica y por lo tanto ya que α es la unica subformula de α, la situacion es facil de probar.

CASO $\alpha = \forall x_i \alpha_1$.
Si $\varphi = \alpha$, entonces la situacion es facil de probar. Si $\varphi \neq \alpha$, entonces la ocurrencia de φ a reemplazar sucede en α_1 y por lo tanto $\overline{\alpha} = \forall x_i \overline{\alpha_1}$. Se tiene entonces que para un $\vec{a}$ dado,

$$\begin{array}{c}
\mathbf{A} \models \alpha[\vec{a}] \\
\Updownarrow \\
\mathbf{A} \models \alpha_1[\downarrow_i^a \vec{a}]\,, \text{ para cada } a \in A \\
\Updownarrow \\
\mathbf{A} \models \overline{\alpha_1}[\downarrow_i^a \vec{a}]\,, \text{ para cada } a \in A \\
\Updownarrow \\
\mathbf{A} \models \overline{\alpha}[\vec{a}]
\end{array}$$

CASO $\alpha = (\alpha_1 \vee \alpha_2)$.
Si $\varphi = \alpha$, entonces la situacion es facil de probar. Supongamos $\varphi \neq \alpha$ y supongamos que la ocurrencia de φ a reemplazar sucede en α_1. Entonces $\overline{\alpha} = (\overline{\alpha_1} \vee \alpha_2)$ y tenemos que

$$\begin{array}{c} \mathbf{A} \models \alpha\,[\vec{a}] \\ \Updownarrow \\ \mathbf{A} \models \alpha_1\,[\vec{a}] \text{ o } \mathbf{A} \models \alpha_2\,[\vec{a}] \\ \Updownarrow \\ \mathbf{A} \models \overline{\alpha_1}\,[\vec{a}] \text{ o } \mathbf{A} \models \alpha_2\,[\vec{a}] \\ \Updownarrow \\ \mathbf{A} \models \overline{\alpha}\,[\vec{a}] \end{array}$$

Los demas casos son dejados al lector.

Dejamos al lector el chequeo de la universalidad del resto de las reglas. ■

Axiomas logicos Recordemos que dada una teoria (Σ, τ), los elementos de Σ son llamados axiomas propios y en general no son sentencias universalmente validas.

En las pruebas formales sera necesario usar ciertas verdades universales y obvias las cuales llamaremos *axiomas logicos*. Mas concretamente, llamaremos *axiomas logicos de tipo* τ a todas las sentencias de alguna de las siguientes formas.

1. $(\varphi \leftrightarrow \varphi)$
2. $(t \equiv t)$
3. $(\varphi \vee \neg\varphi)$
4. $(\varphi \leftrightarrow \neg\neg\varphi)$
5. $(\neg\forall v\psi \leftrightarrow \exists v\neg\psi)$
6. $(\neg\exists v\psi \leftrightarrow \forall v\neg\psi)$

donde $t \in T_c^\tau$, $\varphi \in S^\tau$, $\psi \in F^\tau$, $v \in Var$ y $Li(\psi) \subseteq \{v\}$. Con $AxLog^\tau$ denotaremos el conjunto

$$\{\varphi \in S^\tau : \varphi \text{ es un axioma logico de tipo } \tau\}$$

Notese que hay infinitos axiomas logicos de tipo τ, es decir el conjunto $AxLog^\tau$ es un conjunto infinito de palabras. Por ejemplo, el formato dado en 1. produce una cantidad infinita de axiomas logicos, a saber todas las sentencias de la forma $(\varphi \leftrightarrow \varphi)$, donde φ es cualquier sentencia de tipo τ.

Ejercicio: Pruebe que cada sentencia de $AxLog^\tau$ es universalmente valida

Justificaciones Remitimos a la Seccion 3.3.1 por la definicion del alfabeto Num y la funcion $\overline{\quad} : \omega \to Num^*$ Sea $Nombres_1$ el conjunto formado por las siguientes palabras

EXISTENCIA
COMMUTATIVIDAD
PARTICULARIZACION
ABSURDO
EVOCACION
CONJUNCIONELIMINACION
EQUIVALENCIAELIMINACION
DISJUNCIONINTRODUCCION
ELECCION
GENERALIZACION

Sea $Nombres_2$ el conjunto formado por las siguientes palabras

MODUSPONENS
TRANSITIVIDAD
CONJUNCIONINTRODUCCION
EQUIVALENCIAINTRODUCCION
DISJUNCIONELIMINACION
REEMPLAZO

Una *justificacion basica* es una palabra perteneciente a la union de los siguientes conjuntos de palabras

$$\{\text{CONCLUSION}, \text{AXIOMAPROPIO}, \text{AXIOMALOGICO}\}$$

$$\{\alpha(\bar{k}) : k \in \mathbf{N} \text{ y } \alpha \in Nombres_1\}$$
$$\{\alpha(\bar{j}, \bar{k}) : j, k \in \mathbf{N} \text{ y } \alpha \in Nombres_2\}$$

$$\{\text{DIVISIONPORCASOS}(\bar{j}, \bar{k}, \bar{l}) : j, k, l \in \mathbf{N}\}$$

Usaremos $JustBas$ para denotar el conjunto formado por todas las justificaciones basicas. Una *justificacion* es una palabra que ya sea es una justificacion basica o pertenece a la union de los siguientes conjuntos de palabras

$$\{\text{HIPOTESIS}\bar{k} : k \in \mathbf{N}\}$$

$$\{\text{TESIS}\bar{j}\alpha : j \in \mathbf{N} \text{ y } \alpha \in JustBas\}$$

Usaremos $Just$ para denotar el conjunto formado por todas las justificaciones. Cabe destacar que los elementos de $Just$ son palabras del alfabeto formado por los siguientes simbolos

( ) , 0 1 2 3 4 5 6 7 8 9 A B C D E G H I J L M N O P Q R S T U V X Z

Concatenaciones balanceadas de justificaciones Para construir el concepto de prueba elemental deberiamos trabajar con sucesiones finitas de justificaciones pero el siguiente lema nos dice que podemos reemplazarlas por ciertas palabras, i.e. sus concatenaciones, sin perder informacion.

Lemma 214 *Sea* $\mathbf{J} \in Just^{+}$. *Hay unicos* $n \geq 1$ *y* $J_1, ..., J_n \in Just$ *tales que* $\mathbf{J} = J_1...J_n$.

Proof. Supongamos $J_1, ..., J_n$, $J'_1, ..., J'_m$, con $n, m \geq 1$, son justificaciones tales que $J_1...J_n = J'_1...J'_m$. Es facil ver que entonces tenemos $J_1 = J'_1$, por lo cual $J_2...J_n = J'_2...J'_m$. Un argumento inductivo nos dice que entonces $n = m$ y $J_i = J'_i$, $i = 1, ..., n$ ■

Es decir el lema anterior nos dice que la sucesion $J_1, ..., J_n$ se puede codificar con la palabra $J_1...J_n$ sin perder informacion. Dada $\mathbf{J} \in Just^{+}$, usaremos $n(\mathbf{J})$ y $\mathbf{J}_1, ..., \mathbf{J}_{n(\mathbf{J})}$ para denotar los unicos n y $J_1, ..., J_n$ cuya existencia garantiza el lema anterior.

Dados $i, j \in \omega$, usaremos $\langle i, j \rangle$ para denotar el conjunto $\{l \in \omega : i \leq l \leq j\}$. A los conjuntos de la forma $\langle i, j \rangle$ los llamaremos *bloques*.

Dada $\mathbf{J} \in Just^{+}$ definamos

$$\begin{aligned} \mathcal{B}^{\mathbf{J}} = \{\langle i, j \rangle : 1 \leq i \leq j \leq n(\mathbf{J}) \text{ y} \\ \exists k \; \mathbf{J}_i = \text{HIPOTESIS}\bar{k} \text{ y} \\ \mathbf{J}_j = \text{TESIS}\bar{k}\alpha \text{ para algun } \alpha \in JustBas\} \end{aligned}$$

Diremos que $\mathbf{J} \in Just^{+}$ es *balanceada* si se dan las siguientes

(1) Por cada $k \in \mathbf{N}$ a lo sumo hay un i tal que $\mathbf{J}_i = \text{HIPOTESIS}\bar{k}$ y a lo sumo hay un i tal que $\mathbf{J}_i = \text{TESIS}\bar{k}\alpha$, con $\alpha \in JustBas$

(2) Si $\mathbf{J}_i = \text{HIPOTESIS}\bar{k}$ entonces hay un $l > i$ tal que $\mathbf{J}_l = \text{TESIS}\bar{k}\alpha$, con $\alpha \in JustBas$

(3) Si $\mathbf{J}_i = \text{TESIS}\bar{k}\alpha$, con $\alpha \in JustBas$, entonces hay un $l < i$ tal que $\mathbf{J}_l = \text{HIPOTESIS}\bar{k}$

(4) Si $B_1, B_2 \in \mathcal{B}^{\mathbf{J}}$, entonces $B_1 \cap B_2 = \emptyset$ o $B_1 \subseteq B_2$ o $B_2 \subseteq B_1$

Ejercicio: Supongamos $\mathbf{J} \in Just^+$ es balanceada. Entonces

(a) Si $\langle i,j\rangle \in \mathcal{B}^{\mathbf{J}}$, entonces $i<j$

(b) Si $\langle i,j\rangle, \langle i',j'\rangle \in \mathcal{B}^{\mathbf{J}}$ y $i=i'$, entonces $j=j'$

(c) Si $\langle i,j\rangle, \langle i',j'\rangle \in \mathcal{B}^{\mathbf{J}}$ y $j=j'$, entonces $i=i'$

Pares adecuados Para construir el concepto de prueba elemental deberiamos trabajar con sucesiones finitas de sentencias pero el siguiente lema nos dice que podemos reemplazarlas por ciertas palabras, i.e. sus concatenaciones, sin perder informacion.

Lemma 215 *Sea $\boldsymbol{\varphi} \in S^{\tau+}$. Hay unicos $n \geq 1$ y $\varphi_1,...,\varphi_n \in S^\tau$ tales que $\boldsymbol{\varphi} = \varphi_1...\varphi_n$.*

Proof. Solo hay que probar la unicidad la cual sigue de la Proposicion 186. ∎

Es decir el lema anterior nos dice que la sucesion $\varphi_1,...,\varphi_n$ se puede codificar con la palabra $\varphi_1...\varphi_n$ sin perder informacion. Dada $\boldsymbol{\varphi} \in S^{\tau+}$, usaremos $n(\boldsymbol{\varphi})$ y $\boldsymbol{\varphi}_1,...,\boldsymbol{\varphi}_{n(\boldsymbol{\varphi})}$ para denotar los unicos n y $\varphi_1,...,\varphi_n$ cuya existencia garantiza el lema anterior.

Un *par adecuado de tipo* τ es un par $(\boldsymbol{\varphi},\mathbf{J}) \in S^{\tau+} \times Just^+$ tal que $n(\boldsymbol{\varphi}) = n(\mathbf{J})$ y $\mathbf{J}$ es balanceada.

Sea $(\boldsymbol{\varphi},\mathbf{J})$ un par adecuado de tipo τ. Si $\langle i,j\rangle \in \mathcal{B}^{\mathbf{J}}$, entonces $\boldsymbol{\varphi}_i$ sera la *hipotesis* del bloque $\langle i,j\rangle$ en $(\boldsymbol{\varphi},\mathbf{J})$ y $\boldsymbol{\varphi}_j$ sera la *tesis* del bloque $\langle i,j\rangle$ en $(\boldsymbol{\varphi},\mathbf{J})$. Diremos que $\boldsymbol{\varphi}_i$ *esta bajo la hipotesis* $\boldsymbol{\varphi}_l$ *en* $(\boldsymbol{\varphi},\mathbf{J})$ o que $\boldsymbol{\varphi}_l$ *es una hipotesis de* $\boldsymbol{\varphi}_i$ *en* $(\boldsymbol{\varphi},\mathbf{J})$ cuando haya en $\mathcal{B}^{\mathbf{J}}$ un bloque de la forma $\langle l,j\rangle$ el cual contenga a i. Sean $i,j \in \langle 1,n(\boldsymbol{\varphi})\rangle$. Diremos que i es *anterior* a j *en* $(\boldsymbol{\varphi},\mathbf{J})$ si $i<j$ y ademas para todo $B \in \mathcal{B}^{\mathbf{J}}$ se tiene que $i \in B \Rightarrow j \in B$.

Dependencia de constantes en pares adecuados Sea $(\boldsymbol{\varphi},\mathbf{J})$ un par adecuado de tipo τ. Dadas $e,d \in \mathcal{C}$, diremos que *e depende directamente de d en* $(\boldsymbol{\varphi},\mathbf{J})$ si hay numeros $1 \leq l < j \leq n(\boldsymbol{\varphi})$ tales que

(1) l es anterior a j en $(\boldsymbol{\varphi},\mathbf{J})$

(2) $\mathbf{J}_j = \alpha\text{ELECCION}(\bar{l})$, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$ y $(\boldsymbol{\varphi}_l,\boldsymbol{\varphi}_j) \in Elec^\tau$ via e

(3) d ocurre en $\boldsymbol{\varphi}_l$.

Dados $e,d \in \mathcal{C}$, diremos que *e depende de d en* $(\boldsymbol{\varphi},\mathbf{J})$ si existen $e_0,...,e_{k+1} \in \mathcal{C}$, con $k \geq 0$, tales que

(1) $e_0 = e$ y $e_{k+1} = d$

(2) e_i *depende directamente de* e_{i+1} *en* $(\boldsymbol{\varphi},\mathbf{J})$, para $i=0,...,k$.

Definicion de prueba formal Ahora si estamos en condiciones de definir el concepto de prueba formal en una teoria de primer orden. Sea (Σ, τ) una teoria de primer orden. Sea φ una sentencia de tipo τ. Una *prueba formal de φ en* (Σ, τ) sera un par adecuado $(\boldsymbol{\varphi}, \mathbf{J})$ de algun tipo $\tau_1 = (\mathcal{C} \cup \mathcal{C}_1, \mathcal{F}, \mathcal{R}, a)$, con $\mathcal{C}_1$ finito y disjunto con $\mathcal{C}$, tal que

(1) Cada $\boldsymbol{\varphi}_i$ es una sentencia de tipo τ_1

(2) $\boldsymbol{\varphi}_{n(\boldsymbol{\varphi})} = \varphi$

(3) Si $\langle i, j \rangle \in \mathcal{B}^{\mathbf{J}}$, entonces $\boldsymbol{\varphi}_{j+1} = (\boldsymbol{\varphi}_i \rightarrow \boldsymbol{\varphi}_j)$ y $\mathbf{J}_{j+1} = \alpha$CONCLUSION, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$

(4) Para cada $i = 1, ..., n(\boldsymbol{\varphi})$, se da una de las siguientes

(a) $\mathbf{J}_i =$ HIPOTESIS$\bar{k}$ para algun $k \in \mathbf{N}$

(b) $\mathbf{J}_i = \alpha$CONCLUSION, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$ y hay un j tal que $\langle j, i-1 \rangle \in \mathcal{B}^{\mathbf{J}}$ y $\boldsymbol{\varphi}_i = (\boldsymbol{\varphi}_j \rightarrow \boldsymbol{\varphi}_{i-1})$

(c) $\mathbf{J}_i = \alpha$AXIOMALOGICO, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$ y $\boldsymbol{\varphi}_i$ es un axioma logico de tipo τ_1

(d) $\mathbf{J}_i = \alpha$AXIOMAPROPIO, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$ y $\boldsymbol{\varphi}_i \in \Sigma$

(e) $\mathbf{J}_i = \alpha$PARTICULARIZACION$(\bar{l})$, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$ l anterior a i y $(\boldsymbol{\varphi}_l, \boldsymbol{\varphi}_i) \in Partic^{\tau_1}$

(f) $\mathbf{J}_i = \alpha$COMMUTATIVIDAD$(\bar{l})$, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$, l anterior a i y $(\boldsymbol{\varphi}_l, \boldsymbol{\varphi}_i) \in Commut^{\tau_1}$

(g) $\mathbf{J}_i = \alpha$ABSURDO$(\bar{l})$, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$, l anterior a i y $(\boldsymbol{\varphi}_l, \boldsymbol{\varphi}_i) \in Absur^{\tau_1}$

(h) $\mathbf{J}_i = \alpha$EVOCACION$(\bar{l})$, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$, l anterior a i y $(\boldsymbol{\varphi}_l, \boldsymbol{\varphi}_i) \in Evoc^{\tau_1}$

(i) $\mathbf{J}_i = \alpha$EXISTENCIA$(\bar{l})$, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$, l anterior a i y $(\boldsymbol{\varphi}_l, \boldsymbol{\varphi}_i) \in Exist^{\tau_1}$

(j) $\mathbf{J}_i = \alpha$CONJUNCIONELIMINACION$(\bar{l})$, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$, l anterior a i y $(\boldsymbol{\varphi}_l, \boldsymbol{\varphi}_i) \in ConjElim^{\tau_1}$

(k) $\mathbf{J}_i = \alpha$DISJUNCIONINTRODUCCION$(\bar{l})$, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$, l anterior a i y $(\boldsymbol{\varphi}_l, \boldsymbol{\varphi}_i) \in DisjInt^{\tau_1}$

(l) $\mathbf{J}_i = \alpha$EQUIVALENCIAELIMINACION$(\bar{l})$, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$, l anterior a i y $(\boldsymbol{\varphi}_l, \boldsymbol{\varphi}_i) \in EquivElim^{\tau_1}$

(m) $\mathbf{J}_i = \alpha$MODUSPONENS$(\overline{l_1}, \overline{l_2})$, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$, l_1 y l_2 anteriores a i y $(\boldsymbol{\varphi}_{l_1}, \boldsymbol{\varphi}_{l_2}, \boldsymbol{\varphi}_i) \in ModPon^{\tau_1}$

(n) $\mathbf{J}_i = \alpha$CONJUNCIONINTRODUCCION$(\overline{l_1}, \overline{l_2})$, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$, l_1 y l_2 anteriores a i y $(\boldsymbol{\varphi}_{l_1}, \boldsymbol{\varphi}_{l_2}, \boldsymbol{\varphi}_i) \in ConjInt^{\tau_1}$

(o) $\mathbf{J}_i = \alpha$EQUIVALENCIAINTRODUCCION$(\overline{l_1}, \overline{l_2})$, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$, l_1 y l_2 anteriores a i y $(\boldsymbol{\varphi}_{l_1}, \boldsymbol{\varphi}_{l_2}, \boldsymbol{\varphi}_i) \in EquivInt^{\tau_1}$

(p) $\mathbf{J}_i = \alpha$DISJUNCIONELIMINACION$(\overline{l_1}, \overline{l_2})$, con $\alpha \in \{\varepsilon\} \cup \{$TESIS$\bar{k} : k \in \mathbf{N}\}$, l_1 y l_2 anteriores a i y $(\boldsymbol{\varphi}_{l_1}, \boldsymbol{\varphi}_{l_2}, \boldsymbol{\varphi}_i) \in DisjElim^{\tau_1}$

(q) $\mathbf{J}_i = \alpha$REEMPLAZO$(\overline{l_1}, \overline{l_2})$, con $\alpha \in \{\varepsilon\} \cup \{$TESIS$\bar{k} : k \in \mathbf{N}\}$, l_1 y l_2 anteriores a i y $(\boldsymbol{\varphi}_{l_1}, \boldsymbol{\varphi}_{l_2}, \boldsymbol{\varphi}_i) \in Reemp^{\tau_1}$

(r) $\mathbf{J}_i = \alpha$TRANSITIVIDAD$(\overline{l_1}, \overline{l_2})$, con $\alpha \in \{\varepsilon\} \cup \{$TESIS$\bar{k} : k \in \mathbf{N}\}$, l_1 y l_2 anteriores a i y $(\boldsymbol{\varphi}_{l_1}, \boldsymbol{\varphi}_{l_2}, \boldsymbol{\varphi}_i) \in Trans^{\tau_1}$

(s) $\mathbf{J}_i = \alpha$DIVISIONPORCASOS$(\overline{l_1}, \overline{l_2}, \overline{l_3})$, con $\alpha \in \{\varepsilon\} \cup \{$TESIS$\bar{k} : k \in \mathbf{N}\}$, l_1, l_2 y l_3 anteriores a i y y $(\boldsymbol{\varphi}_{l_1}, \boldsymbol{\varphi}_{l_2}, \boldsymbol{\varphi}_{l_3}, \boldsymbol{\varphi}_i) \in DivPorCas^{\tau_1}$

(t) $\mathbf{J}_i = \alpha$ELECCION$(\bar{l})$, con $\alpha \in \{\varepsilon\} \cup \{$TESIS$\bar{k} : k \in \mathbf{N}\}$, l anterior a i y $(\boldsymbol{\varphi}_l, \boldsymbol{\varphi}_i) \in Elec^{\tau_1}$ via un nombre de cte e, el cual no pertenece a $\mathcal{C}$ y no ocurre en $\boldsymbol{\varphi}_1, ..., \boldsymbol{\varphi}_{i-1}$.

(u) $\mathbf{J}_i = \alpha$GENERALIZACION$(\bar{l})$, con $\alpha \in \{\varepsilon\} \cup \{$TESIS$\bar{k} : k \in \mathbf{N}\}$, l anterior a i y $(\boldsymbol{\varphi}_l, \boldsymbol{\varphi}_i) \in Generaliz^{\tau_1}$ via un nombre de cte c el cual cumple:

    (i) $c \notin \mathcal{C}$

    (ii) c no es un nombre de cte que ocurra en $\boldsymbol{\varphi}$ el cual sea introducido por la aplicacion de la regla de eleccion; es decir para cada $u \in \{1, ..., n(\boldsymbol{\varphi})\}$, si $\mathbf{J}_u = \alpha$ELECCION$(\bar{v})$, con $\alpha \in \{\varepsilon\} \cup \{$TESIS$\bar{k} : k \in \mathbf{N}\}$, entonces no se da que $(\boldsymbol{\varphi}_v, \boldsymbol{\varphi}_u) \in Elec^{\tau_1}$ via c.

    (iii) c no ocurre en ninguna hipotesis de $\boldsymbol{\varphi}_l$.

    (iv) Ningun nombre de constante que ocurra en $\boldsymbol{\varphi}_l$ o en sus hipotesis, depende de c.

A continuacion daremos una prueba formal de $\mu = \forall x_1 \forall x_2((\forall x_3\ x_3 \leq x_1 \wedge \forall x_3\ x_3 \leq x_2) \rightarrow x_1 \equiv x_2)$ en la teoria Po. La idea para hacerla es ir copiando la estructura de la prueba elemental de μ dada al comienso de la Seccion 5. Para facilitar la lectura la escribiremos secuencialmente

1.	$(\forall x_3\ x_3 \leq a \wedge \forall x_3\ x_3 \leq b)$	HIPOTESIS1
2.	$\forall x_3\ x_3 \leq a$	CONJUNCIONELIMINACION(1)
3.	$b \leq a$	PARTICULARIZACION(2)
4.	$\forall x_3\ x_3 \leq b$	CONJUNCIONELIMINACION(1)
5.	$a \leq b$	PARTICULARIZACION(4)
6.	$a \leq b \wedge b \leq a$	CONJUNCIONINTRODUCCION(5, 3)
7.	$\forall x_1 \forall x_2\ ((x_1 \leq x_2 \wedge x_2 \leq x_1) \rightarrow x_1 \equiv x_2)$	AXIOMAPROPIO
8.	$\forall x_2\ ((a \leq x_2 \wedge x_2 \leq a) \rightarrow a \equiv x_2)$	PARTICULARIZACION(7)
9.	$(a \leq b \wedge b \leq a) \rightarrow a \equiv b$	PARTICULARIZACION(8)
10.	$a \equiv b$	TESIS1MODUSPONENS(6, 9)
11.	$(\forall x_3\ x_3 \leq a \wedge \forall x_3\ x_3 \leq b) \rightarrow a \equiv b$	CONCLUSION
12.	$\forall x_2((\forall x_3\ x_3 \leq a \wedge \forall x_3\ x_3 \leq x_2) \rightarrow a \equiv x_2)$	GENERALIZACION(11)
13.	$\forall x_1 \forall x_2((\forall x_3\ x_3 \leq x_1 \wedge \forall x_3\ x_3 \leq x_2) \rightarrow x_1 \equiv x_2)$	GENERALIZACION(12)

pero por supuesto, nuestra prueba formal es en realidad el par $(\boldsymbol{\varphi}, \mathbf{J})$ donde $\boldsymbol{\varphi}$ es la concatenacion de la secuencia de sentencias de arriba y $\mathbf{J}$ es la concatenacion de la secuencia de justificaciones de arriba

6.11.2 El concepto de teorema

Cuando haya una prueba formal de φ en (Σ, τ), diremos que φ es un *teorema* de la teoria (Σ, τ) y escribiremos $(\Sigma, \tau) \vdash \varphi$. A continuacion se dan algunos ejemplos de teoremas de la teoria $(\emptyset, \tau)$.

1. $(\varphi_1 \vee \varphi_2) \rightarrow (\varphi_2 \vee \varphi_1)$ es un teorema de $(\emptyset, \tau)$. Una prueba formal:

1.	$(\varphi_1 \vee \varphi_2)$	HIPOTESIS1
2.	φ_1	HIPOTESIS2
3.	$(\varphi_2 \vee \varphi_1)$	TESIS2DISJUNCIONINTRODUCCION(2)
4.	$(\varphi_1 \rightarrow (\varphi_2 \vee \varphi_1))$	CONCLUSION
5.	φ_2	HIPOTESIS3
6.	$(\varphi_2 \vee \varphi_1)$	TESIS3DISJUNCIONINTRODUCCION(5)
7.	$\varphi_2 \rightarrow (\varphi_2 \vee \varphi_1)$	CONCLUSION
8.	$(\varphi_2 \vee \varphi_1)$	TESIS1DIVISIONPORCASOS$(1, 4, 7)$
9.	$(\varphi_1 \vee \varphi_2) \rightarrow (\varphi_2 \vee \varphi_1)$	CONCLUSION

2. $(\varphi_1 \vee (\varphi_2 \vee \varphi_3)) \rightarrow ((\varphi_1 \vee \varphi_2) \vee \varphi_3)$ es un teorema de $(\emptyset, \tau)$. Una prueba formal:

1.	$(\varphi_1 \vee (\varphi_2 \vee \varphi_3))$	HIPOTESIS1
2.	φ_1	HIPOTESIS2
3.	$(\varphi_1 \vee \varphi_2)$	DISJUNCIONINTRODUCCION(2)
4.	$((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	TESIS2DISJUNCIONINTRODUCCION(3)
5.	$\varphi_1 \rightarrow ((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	CONCLUSION
6.	$(\varphi_2 \vee \varphi_3)$	HIPOTESIS3
7.	φ_2	HIPOTESIS4
8.	$(\varphi_1 \vee \varphi_2)$	DISJUNCIONINTRODUCCION(6)
9.	$((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	TESIS4DISJUNCIONINTRODUCCION(7)
10.	$\varphi_2 \rightarrow ((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	CONCLUSION
11.	φ_3	HIPOTESIS5
12.	$((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	TESIS5DISJUNCIONINTRODUCCION(11)
13.	$\varphi_3 \rightarrow ((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	CONCLUSION
14.	$((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	TESIS3DIVISIONPORCASOS$(6, 10, 13)$
15.	$(\varphi_2 \vee \varphi_3) \rightarrow ((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	CONCLUSION
16.	$((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	TESIS1DIVISIONPORCASOS$(1, 5, 15)$
17.	$(\varphi_1 \vee (\varphi_2 \vee \varphi_3)) \rightarrow ((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	CONCLUSION

3. $((\varphi \wedge (\varphi \vee \psi)) \leftrightarrow \varphi)$ es un teorema de $(\emptyset, \tau)$. Una prueba formal:

1.	$(\varphi \wedge (\varphi \vee \psi))$	HIPOTESIS1
2.	φ	TESIS1CONJUNCIONELIMINACION(1)
3.	$(\varphi \wedge (\varphi \vee \psi)) \rightarrow \varphi$	CONCLUSION
4.	φ	HIPOTESIS2
5.	$(\varphi \vee \psi)$	DISJUNCIONINTRODUCCION(4)
6.	$(\varphi \wedge (\varphi \vee \psi))$	TESIS2CONJUNCIONINTRODUCCION(4, 5)
7.	$\varphi \rightarrow (\varphi \wedge (\varphi \vee \psi))$	CONCLUSION
8.	$((\varphi \wedge (\varphi \vee \psi)) \leftrightarrow \varphi)$	EQUIVALENCIAINTRODUCCION(3, 7)

4. $((\varphi \vee (\varphi \wedge \psi)) \leftrightarrow \varphi)$ es un teorema de $(\emptyset, \tau)$. Una prueba formal:

1.	$(\varphi \vee (\varphi \wedge \psi))$	HIPOTESIS1
2.	φ	HIPOTESIS2
3.	φ	TESIS2EVOCACION(2)
4.	$\varphi \rightarrow \varphi$	CONCLUSION
5.	$(\varphi \wedge \psi)$	HIPOTESIS3
6.	φ	TESIS3CONJUNCIONELIMINACION(5)
7.	$(\varphi \wedge \psi) \rightarrow \varphi$	CONCLUSION
8.	φ	TESIS1DIVISIONPORCASOS(1, 4, 7)
9.	$(\varphi \vee (\varphi \wedge \psi)) \rightarrow \varphi$	CONCLUSION
10.	φ	HIPOTESIS4
11.	$(\varphi \vee (\varphi \wedge \psi))$	TESIS4DISJUNCIONINTRODUCCION(10)
12.	$\varphi \rightarrow (\varphi \vee (\varphi \wedge \psi))$	CONCLUSION
13.	$((\varphi \vee (\varphi \wedge \psi)) \leftrightarrow \varphi)$	EQUIVALENCIAINTRODUCCION(9, 12)

5. $(\varphi \wedge (\varphi_1 \vee \varphi_2)) \rightarrow ((\varphi \wedge \varphi_1) \vee (\varphi \wedge \varphi_2))$ es un teorema de $(\emptyset, \tau)$. Una prueba formal:

1.	$(\varphi \wedge (\varphi_1 \vee \varphi_2))$	HIPOTESIS1
2.	φ	CONJUNCIONELIMINACION(1)
3.	$(\varphi_1 \vee \varphi_2)$	CONJUNCIONELIMINACION(1)
4.	φ_1	HIPOTESIS2
5.	$(\varphi \wedge \varphi_1)$	CONJUNCIONINTRODUCCION(2, 4)
6.	$((\varphi \wedge \varphi_1) \vee (\varphi \wedge \varphi_2))$	TESIS2DISJUNCIONINTRODUCCION(5)
7.	$\varphi_1 \rightarrow ((\varphi \wedge \varphi_1) \vee (\varphi \wedge \varphi_2))$	CONCLUSION
8.	φ_2	HIPOTESIS3
9.	$(\varphi \wedge \varphi_2)$	CONJUNCIONINTRODUCCION(2, 8)
10.	$((\varphi \wedge \varphi_1) \vee (\varphi \wedge \varphi_2))$	TESIS3DISJUNCIONINTRODUCCION(9)
11.	$\varphi_2 \rightarrow ((\varphi \wedge \varphi_1) \vee (\varphi \wedge \varphi_2))$	CONCLUSION
12.	$((\varphi \wedge \varphi_1) \vee (\varphi \wedge \varphi_2))$	TESIS1DIVISIONPORCASOS(3, 7, 11)
13.	$(\varphi \wedge (\varphi_1 \vee \varphi_2)) \rightarrow ((\varphi \wedge \varphi_1) \vee (\varphi \wedge \varphi_2))$	CONCLUSION

6. $((\varphi \wedge \varphi_1) \vee (\varphi \wedge \varphi_2)) \rightarrow (\varphi \wedge (\varphi_1 \vee \varphi_2))$ es un teorema de $(\emptyset, \tau)$. Una prueba

formal:

1.	$((\varphi \wedge \varphi_1) \vee (\varphi \wedge \varphi_2))$	HIPOTESIS1
2.	$(\varphi \wedge \varphi_1)$	HIPOTESIS2
3.	φ	CONJUNCIONELIMINACION(2)
4.	φ_1	CONJUNCIONELIMINACION(2)
5.	$(\varphi_1 \vee \varphi_2)$	DISJUNCIONINTRODUCCION(4)
6.	$\varphi \wedge (\varphi_1 \vee \varphi_2)$	TESIS2CONJUNCIONINTRODUCCION$(3,5)$
7.	$(\varphi \wedge \varphi_1) \to (\varphi \wedge (\varphi_1 \vee \varphi_2))$	CONCLUSION
8.	$(\varphi \wedge \varphi_2)$	HIPOTESIS3
9.	φ	CONJUNCIONELIMINACION(8)
10.	φ_2	CONJUNCIONELIMINACION(8)
11.	$(\varphi_1 \vee \varphi_2)$	DISJUNCIONINTRODUCCION(10)
12.	$\varphi \wedge (\varphi_1 \vee \varphi_2)$	TESIS3CONJUNCIONINTRODUCCION$(9,11)$
13.	$(\varphi \wedge \varphi_2) \to (\varphi \wedge (\varphi_1 \vee \varphi_2))$	CONCLUSION
14.	$(\varphi \wedge (\varphi_1 \vee \varphi_2))$	TESIS1DIVISIONPORCASOS$(1,7,13)$
15.	$((\varphi \wedge \varphi_1) \vee (\varphi \wedge \varphi_2)) \to (\varphi \wedge (\varphi_1 \vee \varphi_2))$	CONCLUSION

7. Sea τ un tipo cualquiera y sea $\varphi =_d \varphi(v)$ una formula de tipo τ. Ya que $(\neg\forall v\varphi \leftrightarrow \exists v\neg\varphi)$ es un axioma logico, tenemos que

$$((\neg\forall v\varphi \leftrightarrow \exists v\neg\varphi), \text{AXIOMALOGICO})$$

es una prueba formal de $(\neg\forall v\varphi \leftrightarrow \exists v\neg\varphi)$ en la teoria $(\emptyset, \tau)$. A continuacion se da una prueba formal en la teoria $(\emptyset, \tau)$ de la sentencia $(\neg\forall v\varphi \leftrightarrow \exists v\neg\varphi)$ la cual no usa el hecho de que $(\neg\forall v\varphi \leftrightarrow \exists v\neg\varphi)$ sea un axioma logico. Notar que en las primeras 10 lineas se prueba $(\neg\exists v\neg\varphi \to \neg\neg\forall v\varphi)$, es decir el contraresiproco de $(\neg\forall v\varphi \to \exists v\neg\varphi)$. De la linea 11 hasta la 17 se prueba $(\neg\forall v\varphi \to \exists v\neg\varphi)$. En las lineas restantes se prueba la implicacion reciproca de $(\neg\forall v\varphi \to \exists v\neg\varphi)$, es decir $(\exists v\neg\varphi \to \neg\forall v\varphi)$ y en el ultimo paso se obtiene $(\neg\forall v\varphi \leftrightarrow \exists v\neg\varphi)$ por la regla de equivalencia-introduccion. Cabe observar que esta prueba formal no es natural u obvia, mas bien es dificil de encontrar.

1.	$\neg\exists v\neg\varphi$	HIPOTESIS1
2.	$\neg\varphi(c)$	HIPOTESIS2
3.	$\exists v\neg\varphi$	EXISTENCIAL(2)
4.	$(\exists v\neg\varphi \wedge \neg\exists v\neg\varphi)$	TESIS2CONJUNCIONINTRODUCCION$(3,1)$
5.	$\neg\varphi(c) \rightarrow (\exists v\neg\varphi \wedge \neg\exists v\neg\varphi)$	CONCLUSION
6.	$\varphi(c)$	ABSURDO(5)
7.	$\forall v\varphi$	GENERALIZACION(6)
8.	$(\forall v\varphi \leftrightarrow \neg\neg\forall v\varphi)$	AXIOMALOGICO
9.	$\neg\neg\forall v\varphi$	TESIS1REEMPLAZO$(7,8)$
10.	$(\neg\exists v\neg\varphi \rightarrow \neg\neg\forall v\varphi)$	CONCLUSION
11.	$\neg\forall v\varphi$	HIPOTESIS3
12.	$\neg\exists v\neg\varphi$	HIPOTESIS4
13.	$\neg\neg\forall v\varphi$	MODUSPONENS$(12,10)$
14.	$(\neg\forall v\varphi \wedge \neg\neg\forall v\varphi)$	TESIS4CONJUNCIONINTRODUCCION$(11,13)$
15.	$\neg\exists v\neg\varphi \rightarrow (\neg\forall v\varphi \wedge \neg\neg\forall v\varphi)$	CONCLUSION
16.	$\exists v\neg\varphi$	TESIS3ABSURDO(15)
17.	$(\neg\forall v\varphi \rightarrow \exists v\neg\varphi)$	CONCLUSION
18.	$\exists v\neg\varphi$	HIPOTESIS5
19.	$\neg\varphi(e)$	ELECCION(18)
20.	$\forall v\varphi$	HIPOTESIS6
21.	$\varphi(e)$	PARTICULARIZACION(20)
22.	$(\varphi(e) \wedge \neg\varphi(e))$	TESIS6CONJUNCIONINTRODUCCION$(21,19)$
23.	$\forall v\varphi \rightarrow (\varphi(e) \wedge \neg\varphi(e))$	CONCLUSION
24.	$\neg\forall v\varphi$	TESIS5ABSURDO(23)
25.	$(\exists v\neg\varphi \rightarrow \neg\forall v\varphi)$	CONCLUSION
26.	$(\neg\forall v\varphi \leftrightarrow \exists v\neg\varphi)$	EQUIVALENCIAINTRODUCCION$(17,25)$

En virtud de la prueba formal anterior se tiene que si redujeramos la lista de axiomas logicos sacando las sentencias de la forma $(\neg\forall v\varphi \leftrightarrow \exists v\neg\varphi)$, el concepto de prueba formal resultante seria equivalente al dado. La razon por la cual se incluyen las sentencias de la forma $(\neg\forall v\varphi \leftrightarrow \exists v\neg\varphi)$ como axiomas logicos es que nuestra definicion de prueba formal en una teoria intenta modelizar o describir en forma matematica a las pruebas reales de los matematicos y la sentencia $(\neg\forall v\varphi \leftrightarrow \exists v\neg\varphi)$ es obviamente cierta para un matematico por lo cual seria un detalle artificioso de nuestra definicion si dicha sentencia resultara dificil de probar con el concepto de prueba formalizado. Dicho de otra forma, nuestro concepto de prueba formal no modelizaria en forma natural a las pruebas matematicas reales. No sucede lo mismo con los axiomas de la forma $(\neg\exists v\varphi \leftrightarrow \forall v\neg\varphi)$ los cuales se pueden probar formalmente en forma directa (y sin usar axiomas de las ultimas dos formas de axiomas logicos) tal como lo haria un matematico. Sin envargo hemos elejido incluir a las sentencias de la forma $(\neg\exists v\varphi \leftrightarrow \forall v\neg\varphi)$ como axiomas logicos por una cuestion estetica y mnemotecnica ya que ambos axiomas $(\neg\forall v\varphi \leftrightarrow \exists v\neg\varphi)$ y $(\neg\exists v\varphi \leftrightarrow \forall v\neg\varphi)$ estan muy emparentados ya que nos dicen como "eliminar la negacion de un cuantificador".

6.11.3 Propiedades basicas de pruebas y teoremas

Por supuesto los numeros asociados a las hipotesis en una prueba son completamente arbitrarios y pueden cambiarse, es decir:

Lemma 216 (Cambio de indice de hipotesis) *Sea* $(\boldsymbol{\varphi}, \mathbf{J})$ *una prueba formal de* φ *en* (Σ, τ)*. Sea* $m \in \mathbf{N}$ *tal que* $\mathbf{J}_i \neq \text{HIPOTESIS}\bar{m}$*, para cada* $i = 1, ..., n(\boldsymbol{\varphi})$*. Supongamos que* $\mathbf{J}_i = \text{HIPOTESIS}\bar{k}$ *y que* $\mathbf{J}_j = \text{TESIS}\bar{k}\alpha$*, con* $[\alpha]_1 \notin Num$*. Sea* $\tilde{\mathbf{J}}$ *el resultado de reemplazar en* $\mathbf{J}$ *la justificacion* $\mathbf{J}_i$ *por* $\text{HIPOTESIS}\bar{m}$ *y reemplazar la justificacion* $\mathbf{J}_j$ *por* $\text{TESIS}\bar{m}\alpha$*. Entonces* $(\boldsymbol{\varphi}, \tilde{\mathbf{J}})$ *es una prueba formal de* φ *en* (Σ, τ)*.*

Tambien podemos cambiar los nombres de cte auxiliares

Lemma 217 (Cambio de ctes auxiliares) *Sea* $(\boldsymbol{\varphi}, \mathbf{J})$ *una prueba formal de* φ *en* (Σ, τ)*. Sea* $\mathcal{C}_1$ *el conjunto de nombres de constante que ocurren en* $\boldsymbol{\varphi}$ *y que no pertenecen a* $\mathcal{C}$*. Sea* $e \in \mathcal{C}_1$*. Sea* $\tilde{e} \notin \mathcal{C} \cup \mathcal{C}_1$ *tal que* $(\mathcal{C} \cup (\mathcal{C}_1 - \{e\}) \cup \{\tilde{e}\}, \mathcal{F}, \mathcal{R}, a)$ *es un tipo. Sea* $\tilde{\boldsymbol{\varphi}}_i =$ *resultado de reemplazar en* $\boldsymbol{\varphi}_i$ *cada ocurrencia de* e *por* $\tilde{e}$*. Entonces* $(\tilde{\boldsymbol{\varphi}}_1 ... \tilde{\boldsymbol{\varphi}}_{n(\boldsymbol{\varphi})}, \mathbf{J})$ *es una prueba formal de* φ *en* (Σ, τ)*.*

Proof. Sean

$$\begin{aligned} \tau_1 &= (\mathcal{C} \cup \mathcal{C}_1, \mathcal{F}, \mathcal{R}, a) \\ \tau_2 &= (\mathcal{C} \cup (\mathcal{C}_1 - \{e\}) \cup \{\tilde{e}\}, \mathcal{F}, \mathcal{R}, a) \end{aligned}$$

Para cada $c \in \mathcal{C} \cup (\mathcal{C}_1 - \{e\})$ definamos $\tilde{c} = c$. Notese que el mapeo $c \to \tilde{c}$ es una biyeccion entre el conjunto de nombres de constante de τ_1 y el conjunto de nombres de cte de τ_2. Para cada $t \in T^{\tau_1}$ sea $\tilde{t} =$ resultado de reemplazar en t cada ocurrencia de c por $\tilde{c}$, para cada $c \in \mathcal{C} \cup \mathcal{C}_1$. Analogamente para una formula $\psi \in F^{\tau_1}$, sea $\tilde{\psi} =$ resultado de reemplazar en ψ cada ocurrencia de c por $\tilde{c}$, para cada $c \in \mathcal{C} \cup \mathcal{C}_1$. Notese que los mapeos $t \to \tilde{t}$ y $\psi \to \tilde{\psi}$ son biyecciones naturales entre T^{τ_1} y T^{τ_2} y entre F^{τ_1} y F^{τ_2}, respectivamente. Notese que cualesquiera sean $\psi_1, \psi_2 \in F^{\tau_1}$, tenemos que ψ_1 se deduce de ψ_2 por la regla de generalizacion con constante c sii $\tilde{\psi}_1$ se deduce de $\tilde{\psi}_2$ por la regla de generalizacion con constante $\tilde{c}$. Para las otras reglas sucede lo mismo. Notese tambien que c ocurre en ψ sii $\tilde{c}$ ocurre en $\tilde{\psi}$. Mas aun notese que c depende de d en $(\boldsymbol{\varphi}, \mathbf{J})$ sii $\tilde{c}$ depende de $\tilde{d}$ en $(\tilde{\boldsymbol{\varphi}}, \mathbf{J})$, donde $\tilde{\boldsymbol{\varphi}} = \widetilde{\boldsymbol{\varphi}_1} ... \widetilde{\boldsymbol{\varphi}_{n(\boldsymbol{\varphi})}}$. Ahora es facil chequear que $(\tilde{\boldsymbol{\varphi}}, \mathbf{J})$ es una prueba formal de φ en (Σ, τ) basandose en que $(\boldsymbol{\varphi}, \mathbf{J})$ es una prueba formal de φ en (Σ, τ). ■

Lemma 218 *Sea* (Σ, τ) *una teoria.*

(1) Si $(\Sigma, \tau) \vdash \varphi_1, ..., \varphi_n$ *y* $(\Sigma \cup \{\varphi_1, ..., \varphi_n\}, \tau) \vdash \varphi$*, entonces* $(\Sigma, \tau) \vdash \varphi$*.*

(2) Si $(\Sigma,\tau)\vdash\varphi_1,...,\varphi_n$ y φ se deduce por alguna regla universal a partir de $\varphi_1,...,\varphi_n$, entonces $(\Sigma,\tau)\vdash\varphi$.

(3) $(\Sigma,\tau)\vdash(\varphi\rightarrow\psi)$ si y solo si $(\Sigma\cup\{\varphi\},\tau)\vdash\psi$.

Proof. (1) Notese que basta con hacer el caso $n=1$. Supongamos entonces que $(\Sigma,\tau)\vdash\varphi_1$ y $(\Sigma\cup\{\varphi_1\},\tau)\vdash\varphi$. Sea $(\alpha_1...\alpha_h, I_1...I_h)$ una prueba formal de φ_1 en (Σ,τ). Sea $(\psi_1...\psi_m, J_1...J_m)$ una prueba formal de φ en $(\Sigma\cup\{\varphi_1\},\tau)$. Notese que por los Lemas 216 y 217 podemos suponer que estas dos pruebas no comparten ningun nombre de constante auxiliar y que tampoco comparten numeros asociados a hipotesis o tesis. Para cada $i=1,...,m$, definamos $\widetilde{J_i}$ de la siguiente manera.

- Si $J_i=\alpha$AXIOMAPROPIO, con $\alpha\in\{\varepsilon\}\cup\{\text{TESIS}\bar{k}:k\in\mathbf{N}\}$ y $\psi_i=\varphi_1$, entonces $\widetilde{J_i}=\alpha$EVOCACION$(\bar{h})$
- Si $J_i=\alpha$AXIOMAPROPIO, con $\alpha\in\{\varepsilon\}\cup\{\text{TESIS}\bar{k}:k\in\mathbf{N}\}$ y $\psi_i\notin\{\varphi_1\}$, entonces $\widetilde{J_i}=\alpha$AXIOMAPROPIO.
- Si $J_i=\alpha$AXIOMALOGICO, con $\alpha\in\{\varepsilon\}\cup\{\text{TESIS}\bar{k}:k\in\mathbf{N}\}$, entonces $\widetilde{J_i}=\alpha$AXIOMALOGICO
- Si $J_i=\alpha$CONCLUSION, con $\alpha\in\{\varepsilon\}\cup\{\text{TESIS}\bar{k}:k\in\mathbf{N}\}$, entonces $\widetilde{J_i}=\alpha$CONCLUSION.
- Si $J_i=\text{HIPOTESIS}\bar{k}$, entonces $\widetilde{J_i}=\text{HIPOTESIS}\bar{k}$
- Si $J_i=\alpha R(\overline{l_1},...,\overline{l_k})$, con $\alpha\in\{\varepsilon\}\cup\{\text{TESIS}\bar{k}:k\in\mathbf{N}\}$, entonces $\widetilde{J_i}=\alpha R(\overline{l_1+h},...,\overline{l_k+h})$

Es facil chequear que

$$(\alpha_1...\alpha_h\psi_1...\psi_m, I_1...I_h\widetilde{J_1}...\widetilde{J_m})$$

es una prueba formal de φ en (Σ,τ)

(2) Supongamos que $(\Sigma,\tau)\vdash\varphi_1,...,\varphi_n$ y que φ se deduce por regla R a partir de $\varphi_1,...,\varphi_n$, con R universal. Notese que

$$\begin{array}{lll}
1. & \varphi_1 & \text{AXIOMAPROPIO} \\
2. & \varphi_2 & \text{AXIOMAPROPIO} \\
\vdots & \vdots & \vdots \\
n. & \varphi_n & \text{AXIOMAPROPIO} \\
n+1. & \varphi & R(\bar{1},...,\bar{n})
\end{array}$$

es una prueba formal de φ en $(\Sigma\cup\{\varphi_1,...,\varphi_n\},\tau)$, lo cual por (1) nos dice que $(\Sigma,\tau)\vdash\varphi$.

(3) Supongamos $(\Sigma,\tau)\vdash(\varphi\rightarrow\psi)$. Entonces tenemos que $(\Sigma\cup\{\varphi\},\tau)\vdash(\varphi\rightarrow\psi),\varphi$, lo cual por (2) nos dice que $(\Sigma\cup\{\varphi\},\tau)\vdash\psi$. Supongamos ahora que $(\Sigma\cup\{\varphi\},\tau)\vdash\psi$. Sea $(\varphi_1...\varphi_n, J_1..., J_n)$ una prueba formal de ψ en $(\Sigma\cup\{\varphi\},\tau)$. Para cada $i=1,...,n$, definamos $\widetilde{J_i}$ de la siguiente manera.

- Si $\varphi_i = \varphi$ y $J_i = \alpha$AXIOMAPROPIO, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$, entonces $\widetilde{J_i} = \alpha$EVOCACION(1)

- Si $\varphi_i \neq \varphi$ y $J_i = \alpha$AXIOMAPROPIO, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$, entonces $\widetilde{J_i} = \alpha$AXIOMAPROPIO

- Si $J_i = \alpha$AXIOMALOGICO, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$, entonces $\widetilde{J_i} = \alpha$AXIOMALOGICO

- Si $J_i = \alpha$CONCLUSION, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$, entonces $\widetilde{J_i} = \alpha$CONCLUSION

- Si $J_i =$ HIPOTESIS$\bar{k}$, entonces $\widetilde{J_i} =$ HIPOTESIS$\bar{k}$

- Si $J_i = \alpha R(\overline{l_1}, ..., \overline{l_k})$, con $\alpha \in \{\varepsilon\} \cup \{\text{TESIS}\bar{k} : k \in \mathbf{N}\}$, entonces $\widetilde{J_i} = \alpha P(\overline{l_1 + 1}, ..., \overline{l_k + 1})$

Sea m tal que ninguna J_i es igual a HIPOTESIS$\bar{m}$. Notese que $\widetilde{J_n}$ no es de la forma TESIS$\bar{k}\beta$ ni de la forma HIPOTESIS$\bar{k}$ (por que?) por lo cual TESIS$\bar{m}\widetilde{J_n}$ es una justificacion. Es facil chequear que

$$(\varphi\varphi_1...\varphi_n(\varphi \to \psi), \text{HIPOTESIS}\bar{m}\widetilde{J_1}...\widetilde{J_{n-1}}\text{TESIS}\bar{m}\widetilde{J_n}\text{CONCLUSION})$$

es una prueba formal de $(\varphi \to \psi)$ en (Σ, τ) ■

6.11.4 Consistencia

Una teoria (Σ, τ) sera *inconsistente* cuando haya una sentencia φ tal que $(\Sigma, \tau) \vdash (\varphi \wedge \neg\varphi)$. Una teoria (Σ, τ) sera *consistente* cuando no sea inconsistente.

Lemma 219 *Sea (Σ, τ) una teoria.*

(1) Si (Σ, τ) es inconsistente, entonces $(\Sigma, \tau) \vdash \varphi$, para toda sentencia φ.

(2) Si (Σ, τ) es consistente y $(\Sigma, \tau) \vdash \varphi$, entonces $(\Sigma \cup \{\varphi\}, \tau)$ es consistente.

(3) Si $(\Sigma, \tau) \nvdash \neg\varphi$, entonces $(\Sigma \cup \{\varphi\}, \tau)$ es consistente.

Proof. (1) Si (Σ, τ) es inconsistente, entonces por definicion tenemos que $(\Sigma, \tau) \vdash \psi \wedge \neg\psi$ para alguna sentencia ψ. Dada una sentencia cualquiera φ tenemos que φ se deduce por la regla del absurdo a partir de $\psi \wedge \neg\psi$ con lo cual (2) del Lema 218 nos dice que $(\Sigma, \tau) \vdash \varphi$

(2) Supongamos (Σ, τ) es consistente y $(\Sigma, \tau) \vdash \varphi$. Si $(\Sigma \cup \{\varphi\}, \tau)$ fuera inconsistente, entonces $(\Sigma \cup \{\varphi\}, \tau) \vdash \psi \wedge \neg\psi$, para alguna sentencia ψ, lo cual por (1) del Lema 218 nos diria que $(\Sigma, \tau) \vdash \psi \wedge \neg\psi$, es decir nos diria que (Σ, τ) es inconsistente.

(3) es dejada al lector. ■

6.11.5 El teorema de correccion

Como ya vimos en las secciones anteriores, el concepto matematico de prueba formal en una teoria (Σ,τ) fue hecho como un intento de modelizar ciertas pruebas que realizan los matematicos profecionales, a las que llamamos pruebas elementales. Es claro que cuando un matematico hace una prueba elemental de una setencia φ en una teoria (Σ,τ), comienza imaginando una estructura $\mathbf{A}$ de tipo τ de la cual lo unico que sabe es que ella satisface todas las sentencias de Σ, y luego al finalizar la prueba concluye que dicho modelo imaginario satisface la ultima sentencia de la prueba. En algun sentido la mision de una prueba es justamente eso: justificar con solidez que la sentencia a probar vale en todos los modelos de la teoria.

O sea que si nuestro concepto de prueba formal permitiera probar sentencias que no sean verdaderas en todos los modelos de la teoria, no seria correcto. Este no es el caso y el teorema que asegura que las pruebas formales solo prueban sentencias verdaderas en todos los modelos se llama Teorema de Correccion. Lo enunciaremos fomalmente a continuacion aunque no daremos la prueba ya que es dificultosa. Antes una definicion. Dada (Σ,τ) una teoria, escribiremos $(\Sigma,\tau)\models\varphi$ cuando φ sea verdadera en todo modelo de (Σ,τ).

Theorem 220 (Teorema de Correccion) $(\Sigma,\tau)\vdash\varphi$ *implica* $(\Sigma,\tau)\models\varphi$*.*

Cabe destacar que el teorema de correccion hace parte de la tarea encomendada en el punto (4) del programa de logica dado en la Seccion 6.6 ya que asegura que nuestro concepto de prueba formal no es demaciado permisivo como para permitir probar sentencias que son falsas en algun modelo de la teoria. Pero dicho concepto podria ser incorrecto en el sentido que podria haber pruebas elementales dadas por matematicos la cuales no puedan ser simuladas por pruebas formales. Por ejemplo podria pasar que mañana un matematico diera una prueba elemental de una sentencia φ en una teoria (Σ,τ) pero que no haya una prueba formal de φ en (Σ,τ). En tal caso nuestro modelo de prueba formal seria un modelo erroneo del concepto de prueba elemental, por ser incompleto. Por supuesto en ese caso podriamos mejorarlo viendo la prueba elemental dada por dicho matematico y enrriquesiendo a la luz de dicha prueba nuestra definicion de prueba formal. De todas maneras nos quedaria la duda de que aun esta nueva definicion de prueba sea incompleta .... Como veremos el Teorema de Completitud de Godel prueba que este no es el caso!

Un corolario muy importante del Teorema de Correccion es el siguiente.

Corollary 221 *Si* (Σ,τ) *tiene un modelo, entonces* (Σ,τ) *es consistente.*

Proof. Supongamos $\mathbf{A}$ es un modelo de (Σ,τ). Si (Σ,τ) fuera inconsistente, tendriamos que hay una $\varphi\in S^t$ tal que $(\Sigma,\tau)\vdash(\varphi\wedge\neg\varphi)$, lo cual por el Teorema de Correccion nos diria que $\mathbf{A}\models(\varphi\wedge\neg\varphi)$ ■

Concluimos la subseccion dando algunos ejemplos que muestran que si hacemos mas permisiva la definicion de prueba formal, esta ya no resulta correcta.

Ejemplo 1: Este ejemplo muestra que en la sentencia a generalizar (dentro de una prueba formal) no puede ocurrir un nombre de cte el cual dependa del nombre de cte a generalizar. Sea $\tau = (\emptyset, \{f^1\}, \emptyset, a)$ y sea $\Sigma = \{\forall y \exists x\ y \equiv f(x)\}$. Sea $T = (\Sigma, \tau)$. Notese que una estructura $\mathbf{A}$ de tipo τ es modelo de T sii $f^{\mathbf{A}}$ es una funcion sobre. Consideremos

1.	$\forall y \exists x\ y \equiv f(x)$	AXIOMAPROPIO
2.	$\exists x\ y_0 \equiv f(x)$	PARTICULARIZACION(1)
3.	$y_0 \equiv f(e)$	ELECCION(2)
4.	$\forall y\ y \equiv f(e)$	GENERALIZACION(3)
5.	$c \equiv f(e)$	PARTICULARIZACION(4)
6.	$d \equiv f(e)$	PARTICULARIZACION(4)
7.	$f(e) \equiv d$	COMMUTATIVIDAD(6)
8.	$c \equiv d$	TRANSITIVIDAD(5, 7)
9.	$\forall y\ c \equiv y$	GENERALIZACION(8)
10.	$\forall x \forall y\ x \equiv y$	GENERALIZACION(9)

Obviamente, si permitieramos que lo anterior fuera una prueba formal, dejaria de valer el teorema de correccion ya que hay muchos modelos de T, los cuales no satisfacen $\forall x \forall y\ x \equiv y$.

Ejemplo 2: El siguiente ejemplo muestra que el nombre de cte a generalizar no puede ocurrir en hipotesis de la sentencia a la cual se le aplica la generalizacion. Sea $\tau = (\{1\}, \emptyset, \emptyset, \emptyset)$ y sea $T = (\emptyset, \tau)$. Consideremos

1.	$c \equiv 1$	HIPOTESIS1
2.	$\forall x\ x \equiv 1$	TESIS1GENERALIZACION(1)
3.	$(c \equiv 1 \rightarrow \forall x\ x \equiv 1)$	CONCLUSION
4.	$\forall y\ (y \equiv 1 \rightarrow \forall x\ x \equiv 1)$	GENERALIZACION(3)
5.	$(1 \equiv 1 \rightarrow \forall x\ x \equiv 1)$	PARTICULARIZACION(4)
6.	$1 \equiv 1$	AXIOMALOGICO
7.	$\forall x\ x \equiv 1$	MODUSPONENS(5, 6)

Si permitieramos que lo anterior fuera una prueba formal, dejaria de valer el teorema de correccion ya que hay muchos modelos de T (toda estructura es un modelo de T) los cuales no satisfacen $\forall x\ x \equiv 1$.

Ejemplo 3: El siguiente ejemplo muestra que la sentencia a generalizar no puede tener una hipotesis en las cual ocurra un nombre de cte que dependa del nombre de cte que se generaliza. Sea $\tau = (\emptyset, \emptyset, \emptyset, \emptyset)$ y sea $T = (\emptyset, \tau)$. Consideremos

1.	$c \equiv c$	AXIOMALOGICO
2.	$\exists z\ z \equiv c$	EXISTENCIA(1)
3.	$e \equiv c$	ELECCION(2)
4.	$d \equiv e$	HIPOTESIS1
5.	$d \equiv c$	TRANSITIVIDAD$(4,3)$
6.	$\forall y\ d \equiv y$	TESIS1GENERALIZACION(5)
7.	$d \equiv e \rightarrow \forall y\ d \equiv y$	CONCLUSION
8.	$\forall x(x \equiv e \rightarrow \forall y\ x \equiv y)$	GENERALIZACION(7)
9.	$e \equiv e \rightarrow \forall y\ e \equiv y$	PARTICULARIZACION(8)
10.	$e \equiv e$	AXIOMALOGICO
11.	$\forall y\ e \equiv y$	MODUSPONENS$(10,9)$
12.	$\forall y\ c \equiv y$	REEMPLAZO$(3,11)$
13.	$\forall x \forall y\ x \equiv y$	GENERALIZACION(12)

6.11.6 El algebra de Lindenbaum

Recordemos que dado un tipo τ, con S^τ denotamos el conjunto de las sentencias de tipo τ, es decir

$$S^\tau = \{\varphi \in F^\tau : Li(\varphi) = \emptyset\}$$

Sea $T = (\Sigma, \tau)$ una teoria. Podemos definir la siguiente relacion binaria sobre S^τ:

$$\varphi \dashv\vdash_T \psi \text{ si y solo si } T \vdash (\varphi \leftrightarrow \psi)$$

Es decir

$$\dashv\vdash_T = \{(\varphi, \psi) \in S^\tau : T \vdash (\varphi \leftrightarrow \psi)\}$$

Lemma 222 *$\dashv\vdash_T$ es una relacion de equivalencia.*

Proof. La relacion es reflexiva ya que $(\varphi \leftrightarrow \varphi)$ es un axioma logico y por lo tanto $((\varphi \leftrightarrow \varphi)$,AXIOMALOGICO$)$ es una prueba formal de $(\varphi \leftrightarrow \varphi)$ en T. Veamos que es simetrica. Supongamos que $\varphi \dashv\vdash_T \psi$, es decir $T \vdash (\varphi \leftrightarrow \psi)$. Ya que $(\psi \leftrightarrow \varphi)$ se deduce de $(\varphi \leftrightarrow \psi)$ por la regla de commutatividad, (2) del Lema 218 nos dice que $T \vdash (\psi \leftrightarrow \varphi)$.

Analogamente, usando la regla de transitividad se puede probar que $\dashv\vdash_T$ es transitiva. ■

Una sentencia φ se dice *refutable en* (Σ, τ) si $(\Sigma, \tau) \vdash \neg\varphi$.

Lemma 223 *Dada una teoria $T = (\Sigma, \tau)$, se tiene que:*

(1) $\{\varphi \in S^\tau : \varphi$ *es un teorema de* $T\} \in S^\tau / \dashv\vdash_T$

(2) $\{\varphi \in S^\tau : \varphi$ *es refutable en* $T\} \in S^\tau / \dashv\vdash_T$

Proof. Haremos la prueba de (2) y dejaremos la prueba de (1) como ejercicio. Sean φ, ψ refutables en T, veremos que $\varphi \dashv\vdash_T \psi$. Notese que

1.	φ	HIPOTESIS1
2.	$\neg\psi$	HIPOTESIS2
3.	$\neg\varphi$	AXIOMAPROPIO
4.	$(\varphi \wedge \neg\varphi)$	TESIS2CONJUNCIONINTRODUCCION$(1,3)$
5.	$\neg\psi \rightarrow (\varphi \wedge \neg\varphi)$	CONCLUSION
6.	ψ	TESIS1ABSURDO(5)
7.	$(\varphi \rightarrow \psi)$	CONCLUSION
8.	ψ	HIPOTESIS3
9.	$\neg\varphi$	HIPOTESIS4
10.	$\neg\psi$	AXIOMAPROPIO
11.	$(\psi \wedge \neg\psi)$	TESIS4CONJUNCIONINTRODUCCION$(8,10)$
12.	$\neg\varphi \rightarrow (\psi \wedge \neg\psi)$	CONCLUSION
13.	φ	TESIS3ABSURDO(12)
14.	$(\psi \rightarrow \varphi)$	CONCLUSION
15.	$(\varphi \leftrightarrow \psi)$	EQUIVALENCIAINTRODUCCION$(7,14)$

justifica que $(\Sigma \cup \{\neg\varphi, \neg\psi\}, \tau) \vdash (\varphi \leftrightarrow \psi)$ lo cual por (1) del Lema 218 nos dice que $(\Sigma, \tau) \vdash (\varphi \leftrightarrow \psi)$, obteniendo que $\varphi \dashv\vdash_T \psi$. Para terminar de probar (2) faltaria ver que si φ es refutable en T y $\varphi \dashv\vdash_T \psi$, entonces ψ es refutable en T. Dejamos al lector la prueba. ■

Dada una teoria $T = (\Sigma, \tau)$ y $\varphi \in S^\tau$, $[\varphi]_T$ denotara la clase de φ con respecto a la relacion de equivalencia $\dashv\vdash_T$. Definiremos sobre $S^\tau/\dashv\vdash_T$ las siguiente operacion binaria s^T:

$$[\varphi]_T \ \mathsf{s}^T \ [\psi]_T = [(\varphi \vee \psi)]_T$$

Una observacion importante es que para que la definicion anterior de la operacion s^T sea inambigua, debemos probar la siguiente propiedad

- Si $[\varphi]_T = [\varphi']_T$ y $[\psi]_T = [\psi']_T$ entonces $[(\varphi \vee \psi)]_T = [(\varphi' \vee \psi')]_T$

Es decir debemos probar que si $T \vdash (\varphi \leftrightarrow \varphi')$ y $T \vdash \left(\psi \leftrightarrow \psi'\right)$, entonces $T \vdash ((\varphi \vee \psi) \leftrightarrow (\varphi' \vee \psi'))$. Pero esto sigue de (1) del Lema 218 ya que

1.	$(\varphi \leftrightarrow \varphi')$	AXIOMAPROPIO
2.	$\left(\psi \leftrightarrow \psi'\right)$	AXIOMAPROPIO
3.	$((\varphi \vee \psi) \leftrightarrow (\varphi \vee \psi))$	AXIOMALOGICO
4.	$((\varphi \vee \psi) \leftrightarrow (\varphi' \vee \psi))$	REEMPLAZO$(1,3)$
5.	$((\varphi \vee \psi) \leftrightarrow (\varphi' \vee \psi'))$	REEMPLAZO$(2,4)$

atestigua que $(\Sigma \cup \{(\varphi \leftrightarrow \varphi'), \left(\psi \leftrightarrow \psi'\right)\}, \tau) \vdash ((\varphi \vee \psi) \leftrightarrow (\varphi' \vee \psi'))$. En forma analoga se puede ver que las siguientes igualdades definen en forma inambigua una operacion binaria i^T sobre $S^\tau/\dashv\vdash_T$ y una operacion unaria ${}^{\mathsf{c}^T}$ sobre $S^\tau/\dashv\vdash_T$:

$$\begin{aligned}[][\varphi]_T \ \mathsf{i}^T \ [\psi]_T &= [(\varphi \wedge \psi)]_T \\ ([\varphi]_T)^{\mathsf{c}^T} &= [\neg\varphi]_T\end{aligned}$$

Dejamos al lector los detalles.

Dada una teoria $T = (\Sigma, \tau)$, denotemos con 1^T al conjunto $\{\varphi \in S^\tau : \varphi$ es un teorema de $T\}$ y con 0^T al conjunto $\{\varphi \in S^\tau : \varphi$ es refutable en $T\}$. Ya vimos en un lema anterior que 0^T y 1^T pertenecen a $S^\tau/\dashv\vdash_T$. Podemos enunciar ahora el siguiente resultado, inspirado en la idea clasica de Boole para el calculo proposicional.

Theorem 224 *Sea* $T = (\Sigma, \tau)$ *una teoria. Entonces* $(S^\tau/\dashv\vdash_T, \mathsf{s}^T, \mathsf{i}^T, {}^{\mathsf{c}^T}, 0^T, 1^T)$ *es un algebra de Boole.*

Proof. Por definicion de algebra de Boole, debemos probar que cualesquiera sean $\varphi_1, \varphi_2, \varphi_3 \in S^\tau$, se cumplen las siguientes igualdades:

(1) $[\varphi_1]_T \ \mathsf{i}^T \ [\varphi_1]_T = [\varphi_1]_T$

(2) $[\varphi_1]_T \ \mathsf{s}^T \ [\varphi_1]_T = [\varphi_1]_T$

(3) $[\varphi_1]_T \ \mathsf{i}^T \ [\varphi_2]_T = [\varphi_2]_T \ \mathsf{i}^T \ [\varphi_1]_T$

(4) $[\varphi_1]_T \ \mathsf{s}^T \ [\varphi_2]_T = [\varphi_2]_T \ \mathsf{s}^T \ [\varphi_1]_T$

(5) $[\varphi_1]_T \ \mathsf{i}^T \ ([\varphi_2]_T \ \mathsf{i}^T \ [\varphi_3]_T) = ([\varphi_1]_T \ \mathsf{i}^T \ [\varphi_2]_T) \ \mathsf{i}^T \ [\varphi_3]_T$

(6) $[\varphi_1]_T \ \mathsf{s}^T \ ([\varphi_2]_T \ \mathsf{s}^T \ [\varphi_3]_T) = ([\varphi_1]_T \ \mathsf{s}^T \ [\varphi_2]_T) \ \mathsf{s}^T \ [\varphi_3]_T$

(7) $[\varphi_1]_T \ \mathsf{s}^T \ ([\varphi_1]_T \ \mathsf{i}^T \ [\varphi_2]_T) = [\varphi_1]_T$

(8) $[\varphi_1]_T \ \mathsf{i}^T \ ([\varphi_1]_T \ \mathsf{s}^T \ [\varphi_2]_T) = [\varphi_1]_T$

(9) $0^T \ \mathsf{s}^T \ [\varphi_1]_T = [\varphi_1]_T$

(10) $[\varphi_1]_T \ \mathsf{s}^T \ 1^T = 1^T$

(11) $[\varphi_1]_T \ \mathsf{s}^T \ ([\varphi_1]_T)^{\mathsf{c}^T} = 1^T$

(12) $[\varphi_1]_T \ \mathsf{i}^T \ ([\varphi_1]_T)^{\mathsf{c}^T} = 0^T$

(13) $[\varphi_1]_T \ \mathsf{i}^T \ ([\varphi_2]_T \ \mathsf{s}^T \ [\varphi_3]_T) = ([\varphi_1]_T \ \mathsf{i}^T \ [\varphi_2]_T) \ \mathsf{s}^T \ ([\varphi_1]_T \ \mathsf{i}^T \ [\varphi_3]_T)$

Veamos por ejemplo que se da (10), es decir probaremos que $[\varphi_1]_T \ \mathsf{s}^T \ 1^T = 1^T$, cualesquiera sea la sentencia φ_1. Ya que $\forall x_1(x_1 \equiv x_1)$ es un teorema de T, atestiguado por la prueba formal

1.	$c \equiv c$	AXIOMALOGICO
2.	$\forall x_1(x_1 \equiv x_1)$	GENERALIZACION(1)

(c es un nombre de cte no perteneciente a $\mathcal{C}$ y tal que $(\mathcal{C} \cup \{c\}, \mathcal{F}, \mathcal{R}, a)$ es un tipo), tenemos que el Lema 223 nos dice que $1^T = \{\varphi \in S^\tau : \varphi$ es un teorema de $T\} = [\forall x_1(x_1 \equiv x_1)]_T$. Es decir que para probar (10) debemos probar que para cualquier $\varphi_1 \in S^\tau$, se da que

$$[\varphi_1]_T \ \mathsf{s}^T \ [\forall x_1(x_1 \equiv x_1)]_T = \{\varphi \in S^\tau : \varphi \text{ es un teorema de } T\}$$

Ya que $[\varphi_1]_T \ \mathsf{s}^T \ [\forall x_1(x_1 \equiv x_1)]_T = [\varphi_1 \vee \forall x_1(x_1 \equiv x_1)]_T$, debemos probar que $\varphi_1 \vee \forall x_1(x_1 \equiv x_1)$ es un teorema de T, lo cual es atestiguado por la siguiente prueba formal

1.	$c \equiv c$	AXIOMALOGICO
2.	$\forall x_1(x_1 \equiv x_1)$	GENERALIZACION(1)
3.	$(\varphi_1 \vee \forall x_1(x_1 \equiv x_1))$	DISJUNCIONINTRODUCCION(2)

Veamos ahora que se da (6), es decir veamos que

$$[\varphi_1]_T \ \mathsf{s}^T \ ([\varphi_2]_T \ \mathsf{s}^T \ [\varphi_3]_T) = ([\varphi_1]_T \ \mathsf{s}^T \ [\varphi_2]_T) \ \mathsf{s}^T \ [\varphi_3]_T$$

cualesquiera sean $\varphi_1, \varphi_2, \varphi_3 \in S^\tau$. Sean $\varphi_1, \varphi_2, \varphi_3 \in S^\tau$ fijas. Por la definicion de la operacion s^T debemos probar que

$$[(\varphi_1 \vee (\varphi_2 \vee \varphi_3))]_T = [((\varphi_1 \vee \varphi_2) \vee \varphi_3)]_T$$

es decir, debemos probar que

$$T \vdash ((\varphi_1 \vee (\varphi_2 \vee \varphi_3)) \leftrightarrow ((\varphi_1 \vee \varphi_2) \vee \varphi_3))$$

Notese que por (2) del Lema 218, basta con probar que

$$\begin{array}{lll} T & \vdash & ((\varphi_1 \vee (\varphi_2 \vee \varphi_3)) \to ((\varphi_1 \vee \varphi_2) \vee \varphi_3)) \\ T & \vdash & (((\varphi_1 \vee \varphi_2) \vee \varphi_3) \to (\varphi_1 \vee (\varphi_2 \vee \varphi_3))) \end{array}$$

La siguiente es una prueba formal de $((\varphi_1 \vee (\varphi_2 \vee \varphi_3)) \to ((\varphi_1 \vee \varphi_2) \vee \varphi_3))$ en T y dejamos al lector la otra prueba formal.

1.	$(\varphi_1 \vee (\varphi_2 \vee \varphi_3))$	HIPOTESIS1
2.	φ_1	HIPOTESIS2
3.	$(\varphi_1 \vee \varphi_2)$	DISJUNCIONINTRODUCCION(2)
4.	$((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	TESIS2DISJUNCIONINTRODUCCION(3)
5.	$\varphi_1 \to ((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	CONCLUSION
6.	$(\varphi_2 \vee \varphi_3)$	HIPOTESIS3
7.	φ_2	HIPOTESIS4
8.	$(\varphi_1 \vee \varphi_2)$	DISJUNCIONINTRODUCCION(6)
9.	$((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	TESIS4DISJUNCIONINTRODUCCION(7)
10.	$\varphi_2 \to ((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	CONCLUSION
11.	φ_3	HIPOTESIS5
12.	$((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	TESIS5DISJUNCIONINTRODUCCION(11)
13.	$\varphi_3 \to ((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	CONCLUSION
14.	$((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	TESIS3DIVISIONPORCASOS$(6, 10, 13)$
15.	$(\varphi_2 \vee \varphi_3) \to ((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	CONCLUSION
16.	$((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	TESIS1DIVISIONPORCASOS$(1, 5, 15)$
17.	$(\varphi_1 \vee (\varphi_2 \vee \varphi_3)) \to ((\varphi_1 \vee \varphi_2) \vee \varphi_3)$	CONCLUSION

El resto de las propiedades pueden ser probadas en forma similar, algunas de las pruebas formales necesarias han sido dadas en los ejemplos que siguen a la definicion de prueba formal ■

Dada una teoria $T = (\Sigma, \tau)$, denotaremos con $\mathcal{A}_T$ al algebra de Boole $(S^\tau / \dashv\vdash_T, \mathsf{s}^T, \mathsf{i}^T, {}^{\mathsf{c}^T}, 0^T, 1^T)$. El algebra $\mathcal{A}_T$ sera llamada el *algebra de Lindenbaum de la teoria* T. El siguiente lema nos da una descripcion agradable del orden parcial asociado al algebra $\mathcal{A}_T$.

Lemma 225 *Sea T una teoria y sea $\leq^T$ el orden parcial asociado al algebra de Boole $\mathcal{A}_T$ (es decir $[\varphi]_T \leq^T [\psi]_T$ si y solo si $[\varphi]_T \; \mathsf{s}^T \; [\psi]_T = [\psi]_T$). Entonces se tiene que*

$$[\varphi]_T \leq^T [\psi]_T \text{ si y solo si } T \vdash (\varphi \to \psi)$$

Proof. Supongamos que $[\varphi]_T \leq^T [\psi]_T$, es decir supongamos que $[\varphi]_T \; \mathsf{s}^T \; [\psi]_T = [\psi]_T$. Por la definicion de s^T tenemos que $[(\varphi \vee \psi)]_T = [\psi]_T$, es decir $T \vdash ((\varphi \vee \psi) \leftrightarrow \psi)$. Es facil ver entonces que $T \vdash (\varphi \to \psi)$. Reciprocamente si $T \vdash (\varphi \to \psi)$, entonces facilmente podemos probar que $T \vdash ((\varphi \vee \psi) \leftrightarrow \psi)$, lo cual nos dice que $[(\varphi \vee \psi)]_T = [\psi]_T$. Por la definicion de s^T tenemos que $[\varphi]_T \; \mathsf{s}^T \; [\psi]_T = [\psi]_T$, lo cual nos dice que $[\varphi]_T \leq^T [\psi]_T$ ■

6.11.7 Teorema de completitud

Hasta el momento tenemos una definicion matematica de prueba formal que modeliza el concepto intuitivo de prueba elemental, el cual corresponde al mundo real de los matematicos profecionales. Ahora bien, nada nos asegura que no aparesca un matematico que realize una prueba elemental de una sentencia φ en una teoria (Σ, τ), y que no haya una prueba formal de φ en (Σ, τ). En tal caso nuestro concepto de prueba seria incompleto (como modelo) aunque, como ya se vio, el mismo es correcto. Esto podria pasar por ejemplo si nos hubiesemos olvidado de incluir en nuestra definicion de prueba formal alguna regla o accion que el matematico usara para probar dicha φ, es decir nos podria pasar que no podamos "traducir" dicha prueba elemental a una prueba formal. Parese dificil poder asegurar o probar que nuestro concepto de prueba formal sea completo en el sentido antes descripto ya que el concepto de prueba elemental es empirico puesto que depende de las acciones de la comunidad matematica profecional y ademas no tiene una formulacion precisa. Por otra parte nada nos asegura que los matematicos profecionales no vayan a descubrir en el futuro algun nuevo "truco" elemental y que nuestro concepto que era completo pase a ser incompleto.

Fue un verdadero desafio cientifico (de los años cercanos a 1900) lidiar con estos problemas, y el teorema de completitud de Godel resuelve todo de una manera limpia y asombrosa. La razon es muy simple: Godel prueba que si una sentencia φ es verdadera en todos los modelos de (Σ, τ), entonces hay una prueba formal de φ en (Σ, τ). Ya que toda prueba elemental que haga un matematico (ahora o en el futuro) siempre probara una sentencia que es verdadera en cada modelo de (Σ, τ), el teorema de Godel nos garantiza que para cada prueba elemental (de ahora y del futuro) habra una prueba formal que pruebe la misma sentencia!

Por supuesto queda la posibilidad de que una prueba elemental dada por algun matematico no sea traducible en forma natural a una prueba formal que pruebe lo mismo (mas alla de que sepamos que hay una gracias a Godel). Sin envargo el lector se ira convenciendo que esto es improbable que suceda, a medida que vaya formalizando distintas pruebas elementales clasicas dadas por los matematicos a lo largo de la historia.

Cabe destacar que entonces el Teorema de Correccion junto con el Teorema de Completitud resuelven el punto (4) del programa de logica dado en la Seccion 6.6.

Para probar el teorema de completitud necesitaremos algunos resultados.

Lemma 226 *Sean* $\tau = (\mathcal{C}, \mathcal{F}, \mathcal{R}, a)$ *y* $\tau' = (\mathcal{C}', \mathcal{F}', \mathcal{R}', a')$ *tipos.*

(1) Si $\mathcal{C} \subseteq \mathcal{C}'$*,* $\mathcal{F} \subseteq \mathcal{F}'$*,* $\mathcal{R} \subseteq \mathcal{R}'$ *y* $a'|_{\mathcal{F}\cup\mathcal{R}} = a$*, entonces* $(\Sigma, \tau) \vdash \varphi$ *implica* $(\Sigma, \tau') \vdash \varphi$

(2) Si $\mathcal{C} \subseteq \mathcal{C}'$*,* $\mathcal{F} = \mathcal{F}'$*,* $\mathcal{R} = \mathcal{R}'$ *y* $a' = a$*, entonces* $(\Sigma, \tau') \vdash \varphi$ *implica* $(\Sigma, \tau) \vdash \varphi$*, cada vez que* $\Sigma \cup \{\varphi\} \subseteq S^\tau$*.*

Proof. (1) Supongamos $(\Sigma, \tau) \vdash \varphi$. Entonces hay una prueba formal $(\varphi_1...\varphi_n, J_1...J_n)$ de φ en (Σ, τ). Notese que aplicando varias veces el Lema 217 podemos obtener una prueba formal $(\tilde{\varphi}_1...\tilde{\varphi}_n, J_1...J_n)$ de φ en (Σ, τ) la cual cumple que si $\mathcal{C}_2$ es el conjunto de nombres de constante que ocurren en $\tilde{\varphi}_1...\tilde{\varphi}_n$ y que no pertenecen a $\mathcal{C}$, entonces:

- $\mathcal{C}_2 \cap \mathcal{C}' = \emptyset$
- $(\mathcal{C}' \cup \mathcal{C}_2, \mathcal{F}', \mathcal{R}', a')$ es un tipo

Pero entonces $(\tilde{\varphi}_1...\tilde{\varphi}_n, J_1...J_n)$ es una prueba formal de φ en (Σ, τ'), con lo cual $(\Sigma, \tau') \vdash \varphi$

(2) Supongamos $(\Sigma, \tau') \vdash \varphi$. Entonces hay una prueba formal $(\boldsymbol{\varphi}, \mathbf{J})$ de φ en (Σ, τ'). Veremos que $(\boldsymbol{\varphi}, \mathbf{J})$ es una prueba formal de φ en (Σ, τ). Ya que $(\boldsymbol{\varphi}, \mathbf{J})$ es una prueba formal de φ en (Σ, τ') hay un conjunto finito $\mathcal{C}_1$, disjunto con $\mathcal{C}'$, tal que $(\mathcal{C}' \cup \mathcal{C}_1, \mathcal{F}, \mathcal{R}, a)$ es un tipo y cada $\boldsymbol{\varphi}_i$ es una sentencia de tipo $(\mathcal{C}' \cup \mathcal{C}_1, \mathcal{F}, \mathcal{R}, a)$. Sea $\widetilde{\mathcal{C}_1} = \mathcal{C}_1 \cup (\mathcal{C}' - \mathcal{C})$. Notese que $\mathcal{C} \cup \widetilde{\mathcal{C}_1} = \mathcal{C}' \cup \mathcal{C}_1$ por lo cual $(\mathcal{C} \cup \widetilde{\mathcal{C}_1}, \mathcal{F}, \mathcal{R}, a)$ es un tipo y cada $\boldsymbol{\varphi}_i$ es una sentencia de tipo $(\mathcal{C} \cup \widetilde{\mathcal{C}_1}, \mathcal{F}, \mathcal{R}, a)$. Esto nos dice que $(\boldsymbol{\varphi}, \mathbf{J})$ cumple (1) de la definicion de prueba formal en (Σ, τ). Todos los otros puntos se cumplen en forma directa, exepto los puntos (4)(t) y (4)(u)(i) para los cuales es necesario notar que $\mathcal{C} \subseteq \mathcal{C}'$. ■

Lemma 227 (Lema del infimo) *Sea* $T = (\Sigma, \tau)$ *una teoria y supongamos que* τ *tiene una cantidad infinita de nombres de cte que no ocurren en las sentencias de* Σ. *Entonces para cada formula* $\varphi =_d \varphi(v)$, *se tiene que en el algebra de Lindembaum* $\mathcal{A}_T$*:*

$$[\forall v\varphi(v)]_T = \inf(\{[\varphi(t)]_T : t \in T_c^\tau\}).$$

Proof. Haremos primero el caso en que v no ocurre acotadamente en φ. Primero notese que $[\forall v\varphi(v)]_T \leq^T [\varphi(t)]_T$, para todo termino cerrado t, ya que podemos dar la siguiente prueba formal:

1.	$\forall v\varphi(v)$	HIPOTESIS1
2.	$\varphi(t)$	TESIS1PARTICULARIZACION(1)
3.	$(\forall v\varphi(v) \to \varphi(t))$	CONCLUSION

Supongamos ahora que $[\psi]_T \leq^T [\varphi(t)]_T$, para todo termino cerrado t. Por hipotesis hay un nombre de cte $c \in \mathcal{C}$ el cual no ocurre en los elementos de $\Sigma \cup \{\psi, \varphi(v)\}$. Ya que $[\psi]_T \leq^T [\varphi(c)]_T$, hay una prueba formal $(\varphi_1 ... \varphi_n, J_1 ... J_n)$ de $(\psi \to \varphi(c))$ en T. Pero entonces es facil de chequear que la siguiente es una prueba formal en $(\Sigma, (\mathcal{C} - \{c\}, \mathcal{F}, \mathcal{R}, a))$ de $(\psi \to \forall v\varphi(v))$:

1.	φ_1	J_1
2.	φ_2	J_2
$\vdots$	$\vdots$	$\vdots$
$n.$	$\varphi_n = (\psi \to \varphi(c))$	J_n
$n+1.$	ψ	HIPOTESIS$\bar{m}$
$n+2.$	$\varphi(c)$	MODUSPONENS$(\overline{n+1}, \bar{n})$
$n+3.$	$\forall v\varphi(v)$	TESIS$\bar{m}$GENERALIZACION$(\overline{n+2})$
$n+4.$	$(\psi \to \forall v\varphi(v))$	CONCLUSION

(con m elejido suficientemente grande y $\tau_1 = \tau$ en la definicion de prueba, es decir con c como el unico nombre de cte auxiliar de la prueba). Por el Lema 226 tenemos entonces que $T \vdash (\psi \to \forall v\varphi(v))$. Cabe destacar que es necesaria la asumsion de que v no ocurra acotadamente en φ para que $(\varphi(c), \forall v\varphi(v)) \in Generaliz^\tau$ (por que?).

Ahora supongamos el caso mas general donde v puede ocurrir acotadamente en φ. Sea $\gamma = \varphi(w)$, donde w es una variable que no ocurre en φ. Claramente $Li(\gamma) \subseteq \{w\}$. Declaremos $\gamma =_d \gamma(w)$. Notese que w no ocurre acotadamente en γ. Por lo ya probado tenemos que

$$[\forall w\gamma(w)]_T = \inf(\{[\gamma(t)]_T : t \in T_c^\tau\})$$

Pero $[\forall v\varphi(v)]_T = [\forall w\gamma(w)]_T$ ya que

1.	$\forall v\varphi(v)$	HIPOTESIS1
2.	$\varphi(c)$ (es igual a $\gamma(c)$)	PARTICULARIZACION(1)
3.	$\forall w\gamma(w)$	TESIS1GENERALIZACION(2)
4.	$(\forall v\varphi(v) \to \forall w\gamma(w))$	CONCLUSION

y

1.	$\forall w\gamma(w)$	HIPOTESIS1
2.	$\varphi(c)$ (es igual a $\gamma(c)$)	PARTICULARIZACION(1)
3.		TESIS1GENERALIZACION(2)
4.	$(\forall v\varphi(v) \to \forall w\gamma(w))$	CONCLUSION

es claro que

$$\begin{array}{rcl} [\forall v\varphi(v)]_T & = & [\forall w\gamma(w)]_T \\ [\varphi(t)]_T & = & [\gamma(t)]_T \text{ para cada } t \in T_c^\tau\}) \end{array}$$

por lo cual obtenemos que

$$[\forall v\varphi(v)]_T = \inf(\{[\varphi(t)]_T : t \in T_c^\tau\}).$$

■

Lemma 228 (Lema de Coincidencia) *Sean τ y τ' dos tipos cualesquiera y sea $\tau_\cap = (\mathcal{C}_\cap, \mathcal{F}_\cap, \mathcal{R}_\cap, a_\cap)$, donde*

$$\begin{array}{rcl} \mathcal{C}_\cap & = & \mathcal{C} \cap \mathcal{C}' \\ \mathcal{F}_\cap & = & \{f \in \mathcal{F} \cap \mathcal{F}' : a(f) = a'(f)\} \\ \mathcal{R}_\cap & = & \{r \in \mathcal{R} \cap \mathcal{R}' : a(r) = a'(r)\} \\ a_\cap & = & a|_{\mathcal{F}_\cap \cup \mathcal{R}_\cap} \end{array}$$

Entonces $\tau_\cap$ es un tipo tal que $T^{\tau_\cap} = T^\tau \cap T^{\tau'}$ y $F^{\tau_\cap} = F^\tau \cap F^{\tau'}$. Sean $\mathbf{A}$ y $\mathbf{A}'$ modelos de tipo τ y τ' respectivamente. Supongamos que $A = A'$ y que $c^{\mathbf{A}} = c^{\mathbf{A}'}$, para cada $c \in \mathcal{C}_\cap$, $f^{\mathbf{A}} = f^{\mathbf{A}'}$, para cada $f \in \mathcal{F}_\cap$ y $r^{\mathbf{A}} = r^{\mathbf{A}'}$, para cada $r \in \mathcal{R}_\cap$.

(a) Para cada $t =_d t(\vec{v}) \in T^{\tau_\cap}$ se tiene que $t^{\mathbf{A}}[\vec{a}] = t^{\mathbf{A}'}[\vec{a}]$, para cada $\vec{a} \in A^n$

(b) Para cada $\varphi =_d \varphi(\vec{v}) \in F^{\tau_\cap}$ se tiene que

$$\mathbf{A} \models \varphi[\vec{a}] \text{ si y solo si } \mathbf{A}' \models \varphi[\vec{a}].$$

(c) Si $\Sigma \cup \{\varphi\} \subseteq S^{\tau_\cap}$, entonces

$$(\Sigma, \tau) \models \varphi \text{ sii } (\Sigma, \tau') \models \varphi.$$

Proof. Dejamos al lector probar que $\tau_\cap$ es un tipo, $T^{\tau_\cap} = T^\tau \cap T^{\tau'}$ y $F^{\tau_\cap} = F^\tau \cap F^{\tau'}$.

(a) y (b) son directos por induccion.

(c) Supongamos que $(\Sigma, \tau) \models \varphi$. Sea $\mathbf{A}'$ un modelo de τ' tal que $\mathbf{A}' \models \Sigma$. Sea $a \in A'$ un elemento fijo. Sea $\mathbf{A}$ el modelo de tipo τ definido de la siguiente manera

- universo de $\mathbf{A} = A'$
- $c^{\mathbf{A}} = c^{\mathbf{A}'}$, para cada $c \in \mathcal{C}_\cap$,
- $f^{\mathbf{A}} = f^{\mathbf{A}'}$, para cada $f \in \mathcal{F}_\cap$
- $r^{\mathbf{A}} = r^{\mathbf{A}'}$, para cada $r \in \mathcal{R}_\cap$
- $c^{\mathbf{A}} = a$, para cada $c \in \mathcal{C} - \mathcal{C}_\cap$
- $f^{\mathbf{A}}(a_1, ..., a_{a(f)}) = a$, para cada $f \in \mathcal{F} - \mathcal{F}_\cap$, $a_1, ..., a_{a(f)} \in A'$
- $r^{\mathbf{A}} = \emptyset$, para cada $r \in \mathcal{R} - \mathcal{R}_\cap$

Ya que $\mathbf{A}' \models \Sigma$, (b) nos dice que $\mathbf{A} \models \Sigma$, lo cual nos dice que $\mathbf{A} \models \varphi$. Nuevamente por (b) tenemos que $\mathbf{A}' \models \varphi$, con lo cual hemos probado que $(\Sigma, \tau') \models \varphi$ ■

Lemma 229 *Sea τ un tipo. Hay una infinitupla $(\gamma_1, \gamma_2, ...) \in F^{\tau\mathbf{N}}$ tal que:*

(1) $\left|Li(\gamma_j)\right| \leq 1$, *para cada* $j = 1, 2, ...$

(2) Si $|Li(\gamma)| \leq 1$, *entonces* $\gamma = \gamma_j$, *para algun* $j \in \mathbf{N}$

Proof. Notese que las formulas de tipo τ son palabras de algun alfabeto finito A. Dado un orden total $\leq$ para A, podemos definir

$$\begin{array}{rcl} \gamma_1 & = & \min_\alpha^{\leq} (\alpha \in F^\tau \wedge |Li(\alpha)| \leq 1) \\ \gamma_{t+1} & = & \min_\alpha^{\leq} (\alpha \in F^\tau \wedge |Li(\alpha)| \leq 1 \wedge (\forall i \in \omega)_{i \leq t} \alpha \neq \gamma_i) \end{array}$$

Notese que para $t \in \mathbf{N}$, tenemos que $\gamma_t = t$-esimo elemento de $\{\alpha \in F^\tau : |Li(\alpha)| \leq 1\}$, con respecto al orden total de A^* inducido por $\leq$. Claramente entonces la infinitupla $(\gamma_1, \gamma_2, ...)$ cumple (1) y (2). ■

Ahora si, el famoso resultado de Godel.

Theorem 230 (Teorema de Completitud) *Sea $T = (\Sigma, \tau)$ una teoria de primer orden. Si $T \models \varphi$, entonces $T \vdash \varphi$.*

Proof. Primero probaremos completitud para el caso en que τ tiene una cantidad infinita de nombres de cte que no ocurren en las sentencias de Σ. Lo probaremos por el absurdo, es decir supongamos que hay una sentencia φ_0 tal que $T \models \varphi_0$ y $T \nvdash \varphi_0$. Notese que ya que $T \nvdash \varphi_0$, tenemos que $[\neg\varphi_0]_T \neq 0^T$. Por el lema anterior hay una infinitupla $(\gamma_1, \gamma_2, ...) \in F^{\tau\mathbf{N}}$ tal que:

- $\left|Li(\gamma_j)\right| \leq 1$, para cada $j = 1, 2, ...$
- Si $|Li(\gamma)| \leq 1$, entonces $\gamma = \gamma_j$, para algun $j \in \mathbf{N}$

Para cada $j \in \mathbf{N}$, sea $w_j \in Var$ tal que $Li(\gamma_j) \subseteq \{w_j\}$. Para cada j, declaremos $\gamma_j =_d \gamma_j(w_j)$. Notese que por el Lema 227 tenemos que $\inf\{[\gamma_j(t)]_T : t \in T_c^\tau\} = [\forall w_j \gamma_j(w_j)]_T$, para cada $j = 1, 2, ...$ Por el Teorema de Rasiova y Sikorski tenemos que hay un filtro primo $\mathcal{U}$ de $\mathcal{A}_T$, el cual cumple:

(a) $[\neg\varphi_0]_T \in \mathcal{U}$

(b) Para cada $j \in \mathbf{N}$, $\{[\gamma_j(t)]_T : t \in T_c^\tau\} \subseteq \mathcal{U}$ implica que $[\forall w_j \gamma_j(w_j)]_T \in \mathcal{U}$

Ya que la infinitupla $(\gamma_1, \gamma_2, ...)$ cubre todas las formulas con a lo sumo una variable libre, podemos reescribir la propiedad (b) de la siguiente manera

(b)$'$ Para cada $\varphi =_d \varphi(v) \in F^\tau$, si $\{[\varphi(t)]_T : t \in T_c^\tau\} \subseteq \mathcal{U}$ entonces $[\forall v \varphi(v)]_T \in \mathcal{U}$

Definamos sobre T_c^τ la siguiente relacion:

$$t \bowtie s \text{ si y solo si } [(t \equiv s)]_T \in \mathcal{U}.$$

Veamos entonces que:

(1) $\bowtie$ es de equivalencia.

(2) Para cada $\varphi =_d \varphi(v_1, ..., v_n) \in F^\tau$, $t_1, ..., t_n, s_1, ..., s_n \in T_c^\tau$, si $t_1 \bowtie s_1$, $t_2 \bowtie s_2$, ..., $t_n \bowtie s_n$, entonces $[\varphi(t_1, ..., t_n)]_T \in \mathcal{U}$ si y solo si $[\varphi(s_1, ..., s_n)]_T \in \mathcal{U}$.

(3) Para cada $f \in \mathcal{F}_n$, $t_1, ..., t_n, s_1, ..., s_n \in T_c^\tau$,

$$t_1 \bowtie s_1, t_2 \bowtie s_2, ..., \ t_n \bowtie s_n \text{ implica } f(t_1, ..., t_n) \bowtie f(s_1, ..., s_n).$$

Probaremos (2). Notese que

$$T \vdash ((t_1 \equiv s_1) \wedge (t_2 \equiv s_2) \wedge ... \wedge (t_n \equiv s_n) \wedge \varphi(t_1, ..., t_n)) \rightarrow \varphi(s_1, ..., s_n)$$

lo cual nos dice que

$$[(t_1 \equiv s_1)]_T \text{ ¡}^T [(t_2 \equiv s_2)]_T \text{ ¡}^T ... \text{ ¡}^T [(t_n \equiv s_n)]_T \text{ ¡}^T [\varphi(t_1, ..., t_n)]_T \leq^T [\varphi(s_1, ..., s_n)]_T$$

de lo cual se desprende que

$$[\varphi(t_1, ..., t_n)]_T \in \mathcal{U} \text{ implica } [\varphi(s_1, ..., s_n)]_T \in \mathcal{U}$$

ya que $\mathcal{U}$ es un filtro. La otra implicacion es analoga.

Para probar (3) podemos tomar $\varphi = (f(v_1, ..., v_n) \equiv f(s_1, ..., s_n))$ y aplicar (2).

Definamos ahora un modelo $\mathbf{A}_\mathcal{U}$ de tipo τ de la siguiente manera:

- Universo de $\mathbf{A}_\mathcal{U} = T_c^\tau / \bowtie$

- $c^{\mathbf{A}_{\mathcal{U}}} = c/\bowtie$, para cada $c \in \mathcal{C}$.

- $f^{\mathbf{A}_{\mathcal{U}}}(t_1/\bowtie, ..., t_n/\bowtie) = f(t_1, ..., t_n)/\bowtie$, para cada $f \in \mathcal{F}_n$, $t_1, ..., t_n \in T_c^\tau$

- $r^{\mathbf{A}_{\mathcal{U}}} = \{(t_1/\bowtie, ..., t_n/\bowtie) : [r(t_1, ..., t_n)]_T \in \mathcal{U}\}$, para cada $r \in \mathcal{R}_n$.

Notese que la definicion de $f^{\mathbf{A}_{\mathcal{U}}}$ es inambigua por (3). Probaremos las siguientes propiedades basicas:

(4) Para cada $t =_d t(v_1, ..., v_n) \in T^\tau$, $t_1, ..., t_n \in T_c^\tau$, tenemos que

$$t^{\mathbf{A}_{\mathcal{U}}}[t_1/\bowtie, ..., t_n/\bowtie] = t(t_1, ..., t_n)/\bowtie$$

(5) Para cada $\varphi =_d \varphi(v_1, ..., v_n) \in F^\tau$, $t_1, ..., t_n \in T_c^\tau$, tenemos que

$$\mathbf{A}_{\mathcal{U}} \models \varphi[t_1/\bowtie, ..., t_n/\bowtie] \text{ si y solo si } [\varphi(t_1, ..., t_n)]_T \in \mathcal{U}.$$

La prueba de (4) es directa por induccion. Probaremos (5) por induccion en el k tal que $\varphi \in F_k^\tau$. El caso $k = 0$ es dejado al lector. Supongamos (5) vale para $\varphi \in F_k^\tau$. Sea $\varphi =_d \varphi(v_1, ..., v_n) \in F_{k+1}^\tau - F_k^\tau$. Hay varios casos:

CASO $\varphi = (\varphi_1 \vee \varphi_2)$.

Notese que por la Convencion Notacional 6, tenemos que $\varphi_i =_d \varphi_i(v_1, ..., v_n)$. Tenemos entonces

$$\begin{array}{c}
\mathbf{A}_{\mathcal{U}} \models \varphi[t_1/\bowtie, ..., t_n/\bowtie] \\
\Updownarrow \\
\mathbf{A}_{\mathcal{U}} \models \varphi_1[t_1/\bowtie, ..., t_n/\bowtie] \text{ o } \mathbf{A}_{\mathcal{U}} \models \varphi_2[t_1/\bowtie, ..., t_n/\bowtie] \\
\Updownarrow \\
[\varphi_1(t_1, ..., t_n)]_T \in \mathcal{U} \text{ o } [\varphi_2(t_1, ..., t_n)]_T \in \mathcal{U} \\
\Updownarrow \\
[\varphi_1(t_1, ..., t_n)]_T \ \mathsf{s}^T \ [\varphi_2(t_1, ..., t_n)]_T \in \mathcal{U} \\
\Updownarrow \\
[(\varphi_1(t_1, ..., t_n) \vee \varphi_2(t_1, ..., t_n))]_T \in \mathcal{U} \\
\Updownarrow \\
[\varphi(t_1, ..., t_n)]_T \in \mathcal{U}.
\end{array}$$

CASO $\varphi = \forall v \varphi_1$, con $v \in Var - \{v_1, ..., v_n\}$. Notese que por la Convencion Notacional 6, tenemos que $\varphi_1 =_d \varphi_1(v_1, ..., v_n, v)$. Tenemos entonces

$$\begin{array}{c}
\mathbf{A}_{\mathcal{U}} \models \varphi[t_1/\bowtie, ..., t_n/\bowtie] \\
\Updownarrow \\
\mathbf{A}_{\mathcal{U}} \models \varphi_1[t_1/\bowtie, ..., t_n/\bowtie, t/\bowtie], \text{ para todo } t \in T_c^\tau \\
\Updownarrow \\
[\varphi_1(t_1, ..., t_n, t)]_T \in \mathcal{U}, \text{ para todo } t \in T_c^\tau \\
\Updownarrow \\
[\forall v \varphi_1(t_1, ..., t_n, v)]_T \in \mathcal{U} \\
\Updownarrow \\
[\varphi(t_1, ..., t_n)]_T \in \mathcal{U}.
\end{array}$$

CASO $\varphi = \exists v\varphi_1$, con $v \in Var - \{v_1, ..., v_n\}$. Notese que por la Convencion Notacional 6, tenemos que $\varphi_1 =_d \varphi_1(v_1, ..., v_n, v)$. Tenemos entonces

$$\begin{gathered}
\mathbf{A}_{\mathcal{U}} \models \varphi[t_1/\bowtie, ..., t_n/\bowtie] \\
\Updownarrow \\
\mathbf{A}_{\mathcal{U}} \models \varphi_1[t_1/\bowtie, ..., t_n/\bowtie, t/\bowtie], \text{ para algun } t \in T_c^\tau \\
\Updownarrow \\
[\varphi_1(t_1, ..., t_n, t)]_T \in \mathcal{U}, \text{ para algun } t \in T_c^\tau \\
\Updownarrow \\
([\varphi_1(t_1, ..., t_n, t)]_T)^{c^T} \notin \mathcal{U}, \text{ para algun } t \in T_c^\tau \\
\Updownarrow \\
[\neg\varphi_1(t_1, ..., t_n, t)]_T \notin \mathcal{U}, \text{ para algun } t \in T_c^\tau \\
\Updownarrow \\
[\forall v\ \neg\varphi_1(t_1, ..., t_n, v)]_T \notin \mathcal{U} \\
\Updownarrow \\
([\forall v\ \neg\varphi_1(t_1, ..., t_n, v)]_T)^{c^T} \in \mathcal{U} \\
\Updownarrow \\
[\neg\forall v\ \neg\varphi_1(t_1, ..., t_n, v)]_T \in \mathcal{U} \\
\Updownarrow \\
[\varphi(t_1, ..., t_n)]_T \in \mathcal{U}.
\end{gathered}$$

Pero ahora notese que (5) en particular nos dice que para cada sentencia $\psi \in S^\tau$, $\mathbf{A}_{\mathcal{U}} \models \psi$ si y solo si $[\psi]_T \in \mathcal{U}$. De esta forma llegamos a que $\mathbf{A}_{\mathcal{U}} \models \Sigma$ y $\mathbf{A}_{\mathcal{U}} \models \neg\varphi_0$, lo cual contradice la suposicion de que $T \models \varphi_0$.

Ahora supongamos que τ es cualquier tipo. Sean s_1 y s_2 un par de simbolos no pertenecientes a la lista

$$\forall\ \exists\ \neg\ \vee\ \wedge\ \rightarrow\ \leftrightarrow\ (\ )\ ,\ \equiv\ \mathsf{X}\ \mathit{0}\ \mathit{1}\ ...\ \mathit{9}\ \mathbf{0}\ \mathbf{1}\ ...\ \mathbf{9}$$

y tales que ninguno ocurra en alguna palabra de $\mathcal{C} \cup \mathcal{F} \cup \mathcal{R}$. Si $T \models \varphi$, entonces usando el Lema de Coincidencia se puede ver que $(\Sigma, (\mathcal{C} \cup \{s_1s_2s_1, s_1s_2s_2s_1, ...\}, \mathcal{F}, \mathcal{R}, a)) \models \varphi$, por lo cual

$$(\Sigma, (\mathcal{C} \cup \{s_1s_2s_1, s_1s_2s_2s_1, ...\}, \mathcal{F}, \mathcal{R}, a)) \vdash \varphi.$$

Pero por Lema 226, tenemos que $T \vdash \varphi$. ■

Corollary 231 *Toda teoria consistente tiene un modelo.*

Proof. Supongamos (Σ, τ) es consistente y no tiene modelos. Entonces $(\Sigma, \tau) \models (\varphi \wedge \neg\varphi)$, con lo cual por completitud $(\Sigma, \tau) \vdash (\varphi \wedge \neg\varphi)$, lo cual es absurdo. ■

Corollary 232 (Teorema de Compacidad) *Sea (Σ, τ) una teoria.*

(a) Si (Σ, τ) es tal que (Σ_0, τ) tiene un modelo, para cada subconjunto finito $\Sigma_0 \subseteq \Sigma$, entonces (Σ, τ) tiene un modelo

(b) Si $(\Sigma, \tau) \models \varphi$, entonces hay un subconjunto finito $\Sigma_0 \subseteq \Sigma$ tal que $(\Sigma_0, \tau) \models \varphi$.

Proof. (a) Veamos que (Σ, τ) es consistente. Supongamos lo contrario, es decir supongamos $(\Sigma, \tau) \vdash (\varphi \wedge \neg\varphi)$, para alguna sentencia φ. Notese que entonces hay un subconjunto finito $\Sigma_0 \subseteq \Sigma$ tal que la teoria $(\Sigma_0, \tau) \vdash (\varphi \wedge \neg\varphi)$ (Σ_0 puede ser formado con los axiomas de Σ usados en una prueba formal que atestigue que $(\Sigma, \tau) \vdash (\varphi \wedge \neg\varphi)$). Pero esto es absurdo ya que por hypotesis dicha teoria (Σ_0, τ) tiene un modelo. O sea que (Σ, τ) es consistente por lo cual tiene un modelo.

(b) Si $(\Sigma, \tau) \models \varphi$, entonces por completitud, $(\Sigma, \tau) \vdash \varphi$. Pero entonces hay un subconjunto finito $\Sigma_0 \subseteq \Sigma$ tal que $(\Sigma_0, \tau) \vdash \varphi$, es decir tal que $(\Sigma_0, \tau) \models \varphi$ (correccion). ■

Interpretacion semantica del algebra de Lindembaum Usando lo teoremas de correccion y completitud podemos dar una representacion semantica del algebra de Lindembaum. Sea $T = (\Sigma, \tau)$ una teoria. Dada $\varphi \in S^\tau$ definamos

$$\mathrm{Mod}_T(\varphi) = \{\mathbf{A} : \mathbf{A} \text{ es modelo de } T \text{ y } \mathbf{A} \vDash \varphi\}$$

Definamos tambien

$$\mathrm{Mod}_T = \{\mathbf{A} : \mathbf{A} \text{ es modelo de } T\}$$

Dado $S \subseteq \mathrm{Mod}_T$ definamos

$$S^c = \mathrm{Mod}_T - S$$

Lemma 233 *$\{\mathrm{Mod}_T(\varphi) : \varphi \in S^\tau\}$ es un subuniverso del algebra de Boole $(\mathcal{P}(\mathrm{Mod}_T), \cup, \cap, ^c, \emptyset, \mathrm{Mod}_T)$*

Proof. Notese que

$$\begin{array}{rcl}
\emptyset & = & \mathrm{Mod}_T(\exists x_1 \neg(x_1 \equiv x_1)) \\
\mathrm{Mod}_T & = & \mathrm{Mod}_T(\forall x_1 (x_1 \equiv x_1)) \\
\mathrm{Mod}_T(\varphi) \cap \mathrm{Mod}_T(\psi) & = & \mathrm{Mod}_T((\varphi \wedge \psi)) \\
\mathrm{Mod}_T(\varphi) \cup \mathrm{Mod}_T(\psi) & = & \mathrm{Mod}_T((\varphi \vee \psi)) \\
\mathrm{Mod}_T(\varphi)^c & = & \mathrm{Mod}_T(\neg\varphi)
\end{array}$$

■

Lemma 234 *Dadas $\varphi, \psi \in S^\tau$ se tiene:*

(1) $[\varphi]_T \leq^T [\psi]_T$ *sii* $\mathrm{Mod}_T(\varphi) \subseteq \mathrm{Mod}_T(\psi)$

(2) $[\varphi]_T = [\psi]_T$ *sii* $\mathrm{Mod}_T(\varphi) = \mathrm{Mod}_T(\psi)$

(3) $[\varphi]_T <^T [\psi]_T$ *sii* $\mathrm{Mod}_T(\varphi) \subsetneq \mathrm{Mod}_T(\psi)$

Proof. (1) Dejamos al lector justificar las siguientes equivalencias:

$$[\varphi]_T \leq^T [\psi]_T \text{ sii } T \vdash (\varphi \rightarrow \psi) \text{ sii } T \models (\varphi \rightarrow \psi) \text{ sii } \mathrm{Mod}_T(\varphi) \subseteq \mathrm{Mod}_T(\psi)$$

(2) y (3) siguen de (1) ■

Ya que $\{\mathrm{Mod}_T(\varphi) : \varphi \in S^\tau\}$ es un subuniverso de $(\mathcal{P}(\mathrm{Mod}_T), \cup, \cap, ^c, \emptyset, \mathrm{Mod}_T)$, tenemos que $(\{\mathrm{Mod}_T(\varphi) : \varphi \in S^\tau\}, \cup, \cap, ^c, \emptyset, \mathrm{Mod}_T)$ es un algebra de Boole. Notese que (2) del lema anterior nos asegura que

$$\begin{array}{rcl} S^\tau/\dashv\vdash_T & \rightarrow & \{\mathrm{Mod}_T(\varphi) : \varphi \in S^\tau\} \\ [\varphi]_T & \rightarrow & \mathrm{Mod}_T(\varphi) \end{array}$$

define en forma inhambigua una funcion. Tenemos entonces el siguiente

Theorem 235 *La funcion*

$$\begin{array}{rcl} S^\tau/\dashv\vdash_T & \rightarrow & \{\mathrm{Mod}_T(\varphi) : \varphi \in S^\tau\} \\ [\varphi]_T & \rightarrow & \mathrm{Mod}_T(\varphi) \end{array}$$

es un isomorfismo de $\mathcal{A}_T$ *en* $(\{\mathrm{Mod}_T(\varphi) : \varphi \in S^\tau\}, \cup, \cap, ^c, \emptyset, \mathrm{Mod}_T)$.

Proof. Llamemosle f a la funcion del enunciado. Es claro que f es sobre. Ademas (2) del lema anterior nos dice que f es inyectiva. Ademas usando las igualdades de la prueba del Lema 233, facilmente podemos ver que f es un homomorfismo por lo cual es un isomorfismo. ■

6.11.8 Teorias completas

Una teoria (Σ, τ) sera llamada *completa* cuando para cada $\varphi \in S^\tau$ se de que $(\Sigma, \tau) \vdash \varphi$ o $(\Sigma, \tau) \vdash \neg\varphi$. Es poco frecuente que una teoria consistente sea completa y esto lo veremos claro despues del siguiente resultado.

Proposition 236 *Sea* (Σ, τ) *una teoria consistente. Son equivalentes*

(1) (Σ, τ) *es completa*

(2) Todos los modelos de (Σ, τ) *son elementalmente equivalentes*

(3) Hay una estructura $\mathbf{A}$ *tal que los teoremas de* (Σ, τ) *son exactamente las sentencias verdaderas en* $\mathbf{A}$

(4) $\mathcal{A}_{(\Sigma,\tau)}$ *tiene exactamente dos elementos*

Proof. Supongamos vale (1). Probaremos (2). Supongamos $\mathbf{A}, \mathbf{B}$ son modelos de (Σ, τ) y supongamos $\mathbf{A} \models \varphi$. Entonces no puede darse $(\Sigma, \tau) \vdash \neg\varphi$ (use correccion) por lo cual tenemos que $(\Sigma, \tau) \vdash \varphi$. Ya que $\mathbf{B}$ es modelo de (Σ, τ) tenemos entonces que $\mathbf{B} \models \varphi$. Esto prueba que $\mathbf{A}$ y $\mathbf{B}$ son elementalmente equivalentes.

Ahora supongamos vale (2). Probaremos (3). Ya que (Σ,τ) es consistente, tiene al menos un modelo. Sea $\mathbf{A}$ uno de ellos. Por correccion todo teorema es verdadero en $\mathbf{A}$. Reciprocamente si $\mathbf{A} \models \varphi$, entonces, por (2), todo modelo de (Σ,τ) satisface φ, lo cual nos dice que φ es un teorema.

Obviamente (3) impica que toda sentencia es o un teorema o refutable y esto nos dice que $0^{(\Sigma,\tau)} \cup 1^{(\Sigma,\tau)} = S^\tau$. Ya que $0^{(\Sigma,\tau)} \cap 1^{(\Sigma,\tau)} = \emptyset$ puesto que (Σ,τ) es consistente, tenemos que vale (4).

Es trivial que (4) implica (1). ■

Ya que lo mas comun es que una teoria tenga un par de modelos no elementalmente equivalentes, la mayoria de las teorias no son completas.

6.11.9 La aritmetica de Peano

En esta seccion desarrollaremos las propiedades basicas de $Arit$, una teoria de primer orden la cual modeliza a la aritmetica. Esta teoria ha sido paradigmatica en el desarrollo de la logica.

Sea $\tau_A = (\{0,1\},\{+^2,.^2\},\{\leq^2\},a)$. Denotemos con $\boldsymbol{\omega}$ a la estructura de tipo τ_A que tiene a ω como universo e interpreta los nombres de τ_A en la manera usual, es decir

$$\begin{array}{l} 0^{\boldsymbol{\omega}} = 0 \\ 1^{\boldsymbol{\omega}} = 1 \\ \leq^{\boldsymbol{\omega}} = \{(n,m) \in \omega^2 : n \leq m\} \\ +^{\boldsymbol{\omega}}(n,m) = n+m, \text{ para cada } n,m \in \omega \\ .^{\boldsymbol{\omega}}(n,m) = n.m, \text{ para cada } n,m \in \omega \end{array}$$

Sea Σ el conjunto formado por las siguientes sentencias:

1. $\forall x_1 \forall x_2 \forall x_3\ x_1 + (x_2 + x_3) \equiv (x_1 + x_2) + x_3$
2. $\forall x_1 \forall x_2\ x_1 + x_2 \equiv x_2 + x_1$
3. $\forall x_1 \forall x_2 \forall x_3\ x_1.(x_2.x_3) \equiv (x_1.x_2).x_3$
4. $\forall x_1 \forall x_2\ x_1.x_2 \equiv x_2.x_1$
5. $\forall x_1\ x_1 + 0 \equiv x_1$
6. $\forall x_1\ x_1.0 \equiv 0$
7. $\forall x_1\ x_1.1 \equiv x_1$
8. $\forall x_1 \forall x_2 \forall x_3\ x_1.(x_2 + x_3) \equiv (x_1.x_2) + (x_1.x_3)$
9. $\forall x_1 \forall x_2 \forall x_3\ (x_1 + x_3 \equiv x_2 + x_3 \rightarrow x_1 \equiv x_2)$
10. $\forall x_1\ x_1 \leq x_1$

11. $\forall x_1 \forall x_2 \forall x_3\ ((x_1 \leq x_2 \wedge x_2 \leq x_3) \rightarrow x_1 \leq x_3)$

12. $\forall x_1 \forall x_2\ ((x_1 \leq x_2 \wedge x_2 \leq x_1) \rightarrow x_1 \equiv x_2)$

13. $\forall x_1 \forall x_2\ (x_1 \leq x_2 \vee x_2 \leq x_1)$

14. $\forall x_1 \forall x_2\ (x_1 \leq x_2 \leftrightarrow \exists x_3\ x_2 \equiv x_1 + x_3)$

15. $0 < 1$

Es facil ver que todas estas sentencias son satisfechas por $\boldsymbol{\omega}$ por lo cual $\boldsymbol{\omega}$ es un modelo de la teoria (Σ, τ_A). Definamos

$$Verd_{\boldsymbol{\omega}} = \{\varphi \in S^{\tau_A} : \boldsymbol{\omega} \models \varphi\}$$

Es claro que todo teorema de (Σ, τ_A) pertenece a $Verd_{\boldsymbol{\omega}}$ (por que?). Un pregunta interesante es si toda sentencia $\varphi \in Verd_{\boldsymbol{\omega}}$ es un teorema de (Σ, τ_A), es decir puede ser probada en forma elemental partiendo de los axiomas de Σ. La respuesta es no y lo explicaremos a continuacion. Sea $\mathbf{Q}^{\geq 0}$ la estructura de tipo τ_A que tiene a $\{r \in \mathbf{Q} : r \geq 0\}$ como universo e interpreta los nombres de τ_A en la manera usual. Note que $\mathbf{Q}^{\geq 0}$ tambien es un modelo de (Σ, τ_A). Pero entonces todo teorema de (Σ, τ_A) debe ser verdadero en $\mathbf{Q}^{\geq 0}$. Pero la sentencia $\forall x\ (x \leq 1 \rightarrow (x \equiv 0 \vee x \equiv 1))$ es falsa en $\mathbf{Q}^{\geq 0}$ por lo cual no es un teorema de (Σ, τ_A) y sin envargo pertenece a $Verd_{\boldsymbol{\omega}}$. Es decir los axiomas de Σ son demaciado generales y deberiamos agregarle axiomas que sean mas caracteristicos de la estructura particular de $\boldsymbol{\omega}$. En esa direccion, a continuacion extenderemos el conjunto Σ con axiomas que nos permitiran hacer pruebas por induccion tal como se lo hace en la aritmetica basica.

Dada una formula $\psi \in F^{\tau_A}$ y variables $v_1, ..., v_{n+1}$, con $n \geq 0$, tales que $Li(\psi) \subseteq \{v_1, ..., v_{n+1}\}$ y $v_i \neq v_j$ siempre que $i \neq j$, denotaremos con $Ind_{\psi, v_1, ..., v_{n+1}}$ a la siguiente sentencia de tipo τ_A

$$\forall v_1 ... \forall v_n\ ((\psi(\vec{v}, 0) \wedge \forall v_{n+1}\ (\psi(\vec{v}, v_{n+1}) \rightarrow \psi(\vec{v}, +(v_{n+1}, 1)))) \rightarrow \forall v_{n+1}\ \psi(\vec{v}, v_{n+1}))$$

donde suponemos que hemos declarado $\psi =_d \psi(v_1, ..., v_{n+1})$. Notese que si por ejemplo $Li(\psi) \subseteq \{x_1, x_2, x_3\}$, entonces las seis sentencias

$$Ind_{\psi, x_1, x_2, x_3}\ \ Ind_{\psi, x_1, x_3, x_2}\ \ Ind_{\psi, x_2, x_1, x_3}\ \ Ind_{\psi, x_2, x_3, x_1}\ \ Ind_{\psi, x_3, x_1, x_2}\ \ Ind_{\psi, x_3, x_2, x_1}$$

son todas distintas.

Sea Σ_A el conjunto que resulta de agregarle a Σ todas las sentencias de la forma $Ind_{\psi, v_1, ..., v_{n+1}}$. Notese que el conjunto Σ_A es infinito.

La teoria (Σ_A, τ_A) sera llamada *Aritmetica de Peano* y la denotaremos con *Arit*. Es intuitivamente claro que

Lemma 237 $\boldsymbol{\omega}$ *es un modelo de Arit*

Proof. Sean $\psi \in F^{\tau_A}$ y $v_1, ..., v_{n+1}$, con $n \geq 0$, tales que $Li(\psi) \subseteq \{v_1, ..., v_{n+1}\}$ y $v_i \neq v_j$ siempre que $i \neq j$. Veremos que $\boldsymbol{\omega} \vDash Ind_{\psi, v_1, ..., v_{n+1}}$. Declaremos $\psi =_d \psi(v_1, ..., v_n, v_{n+1})$. Sea

$$\varphi = ((\psi(\vec{v}, 0) \wedge \forall v_{n+1}\ (\psi(\vec{v}, v_{n+1}) \to \psi(\vec{v}, +(v_{n+1}, 1)))) \to \forall v_{n+1}\ \psi(\vec{v}, v_{n+1}))$$

Declaremos $\varphi =_d \varphi(v_1, ..., v_n)$. Notese que $\boldsymbol{\omega} \vDash Ind_{\psi, v_1, ..., v_{n+1}}$ si y solo si para cada $a_1, ..., a_n \in \omega$ se tiene que $\boldsymbol{\omega} \vDash \varphi[\vec{a}]$. Sean $a_1, ..., a_n \in \omega$ fijos. Probaremos que $\boldsymbol{\omega} \vDash \varphi[\vec{a}]$. Notar que si

$$\boldsymbol{\omega} \nvDash (\psi(\vec{v}, 0) \wedge \forall v_{n+1}\ (\psi(\vec{v}, v_{n+1}) \to \psi(\vec{v}, +(v_{n+1}, 1)))[\vec{a}]$$

entonces $\boldsymbol{\omega} \vDash \varphi[\vec{a}]$ por lo cual podemos hacer solo el caso en que

$$\boldsymbol{\omega} \vDash (\psi(\vec{v}, 0) \wedge \forall v_{n+1}\ (\psi(\vec{v}, v_{n+1}) \to \psi(\vec{v}, +(v_{n+1}, 1)))[\vec{a}]$$

Para probar que $\boldsymbol{\omega} \vDash \varphi[\vec{a}]$, deberemos probar entonces que $\boldsymbol{\omega} \vDash \forall v_{n+1}\ \psi(\vec{v}, v_{n+1})[\vec{a}]$. Sea $S = \{a \in \omega : \boldsymbol{\omega} \vDash \psi(\vec{v}, v_{n+1})[\vec{a}, a]\}$. Ya que $\boldsymbol{\omega} \vDash \psi(\vec{v}, 0)[\vec{a}]$, es facil ver usando el lema de reemplazo que $\boldsymbol{\omega} \vDash \psi(\vec{v}, v_{n+1})[\vec{a}, 0]$, lo cual nos dice que $0 \in S$. Ya que $\boldsymbol{\omega} \vDash \forall v_{n+1}\ (\psi(\vec{v}, v_{n+1}) \to \psi(\vec{v}, +(v_{n+1}, 1))[\vec{a}]$, tenemos que

(1) Para cada $a \in \omega$, si $\boldsymbol{\omega} \vDash \psi(\vec{v}, v_{n+1})[\vec{a}, a]$, entonces $\boldsymbol{\omega} \vDash \psi(\vec{v}, +(v_{n+1}, 1))[\vec{a}, a]$.

Pero por el lema de reemplazo, tenemos que $\boldsymbol{\omega} \vDash \psi(\vec{v}, +(v_{n+1}, 1))[\vec{a}, a]$ sii $\boldsymbol{\omega} \vDash \psi(\vec{v}, v_{n+1})[\vec{a}, a+1]$, lo cual nos dice que

(2) Para cada $a \in \omega$, si $\boldsymbol{\omega} \vDash \psi(\vec{v}, v_{n+1})[\vec{a}, a]$, entonces $\boldsymbol{\omega} \vDash \psi(\vec{v}, v_{n+1})[\vec{a}, a+1]$.

Ya que $0 \in S$ y (2) nos dice que $a \in S$ implica $a+1 \in S$, tenemos que $S = \omega$. Es decir que para cada $a \in \omega$, se da que $\boldsymbol{\omega} \vDash \psi(\vec{v}, v_{n+1})[\vec{a}, a]$ lo cual nos dice que $\boldsymbol{\omega} \vDash \forall v_{n+1}\ \psi(\vec{v}, v_{n+1})[\vec{a}]$.

Es rutina probar que $\boldsymbol{\omega}$ satisface los otros 15 axiomas de *Arit*. ■

El modelo $\boldsymbol{\omega}$ es llamado el *modelo standard* de *Arit*.

Ejercicio: Pruebe que $\mathbf{Q}^{\geq 0}$ no es un modelo de *Arit*, dando una "propiedad inductiva que no cumpla"

Definamos el mapeo $\widehat{\ }: \omega \to \{(\ )\ ,\ +\ 0\ 1\}^*$ de la siguiente manera

$$\begin{array}{rcl} \widehat{0} & = & 0 \\ \widehat{1} & = & 1 \\ \widehat{n+1} & = & +(\widehat{n}, 1), \text{ para cada } n \geq 1 \end{array}$$

Proposition 238 *Hay un modelo de Arit el cual no es isomorfo a* $\boldsymbol{\omega}$

Proof. Sea $\tau = (\{0, 1, \blacktriangle\}, \{+^2, .^2\}, \{\leq^2\}, a)$ y sea $\Sigma = \Sigma_A \cup \{\neg(\widehat{n} \equiv \blacktriangle) : n \in \omega\}$. Por el Teorema de Compacidad la teoria (Σ, τ) tiene un modelo $\mathbf{A} = (A, i)$. Ya que

$$\mathbf{A} \vDash \neg(\widehat{n} \equiv \blacktriangle), \text{ para cada } n \in \omega$$

tenemos que

$$i(\blacktriangle) \neq \widehat{n}^{\mathbf{A}}, \text{ para cada } n \in \omega$$

Por el Lema de Coincidencia la estructura $\mathbf{B} = (A, i|_{\{0,1,+,.,\leq\}})$ es un modelo de *Arit*. Ademas dicho lema nos garantiza que $\widehat{n}^{\mathbf{B}} = \widehat{n}^{\mathbf{A}}$, para cada $n \in \omega$, por lo cual tenemos que

$$i(\blacktriangle) \neq \widehat{n}^{\mathbf{B}}, \text{ para cada } n \in \omega$$

Veamos que $\mathbf{B}$ no es isomorfo a $\boldsymbol{\omega}$. Supongamos $F : \omega \to A$ es un isomorfismo de $\boldsymbol{\omega}$ en $\mathbf{B}$. Es facil de probar por induccion en n que $F(n) = \widehat{n}^{\mathbf{B}}$, para cada $n \in \omega$. Pero esto produce un absurdo ya que nos dice que $i(\blacktriangle)$ no esta en la imagen de F. $\blacksquare$

Ejercicio: Dado un modelo $\mathbf{A}$ de *Arit* y elementos $a, b \in A$, diremos que *a divide a b en* $\mathbf{A}$ cuando haya un $c \in A$ tal que $b = .^{\mathbf{A}}(c, a)$. Un elemento $a \in A$ sera llamado *primo en* $\mathbf{A}$ si $a \neq 1^{\mathbf{A}}$ y sus unicos divisores son $1^{\mathbf{A}}$ y a. Pruebe que hay un modelo de *Arit*, $\mathbf{A}$, en el cual hay infinitos primos no pertenecientes a $\{\widehat{n}^{\mathbf{A}} : n \in \omega\}$.

Lemma 239 *Las siguientes sentencias son teoremas de la aritmetica de Peano:*

(1) $\forall x\ 0 \leq x$

(2) $\forall x\ (x \leq 0 \to x \equiv 0)$

(3) $\forall x \forall y\ (x + y \equiv 0 \to (x \equiv 0 \wedge y \equiv 0))$

(4) $\forall x\ (\neg(x \equiv 0) \to \exists z\ (x \equiv z + 1))$

(5) $\forall x \forall y\ (x < y \to x + 1 \leq y)$

(6) $\forall x \forall y\ (x < y + 1 \to x \leq y)$

(7) $\forall x \forall y\ (x \leq y + 1 \to (x \leq y \vee x \equiv y + 1))$

(8) $\forall x \forall y\ ((x \leq y \wedge y \leq x + 1) \to (x \equiv y \vee x \equiv y + 1))$ *(use (7))*

(9) $\forall x \forall y\ (\neg y \equiv 0 \to \exists q \exists r\ x \equiv q.y + r \wedge r < y)$

Proof. (1) es dejada al lector.

(2)

1.	$x_0 \leq 0$	HIPOTESIS1
2.	$\forall x\ 0 \leq x$	TEOREMA
3.	$0 \leq x_0$	PARTICULARIZACION(2)
4.	$x_0 \leq 0 \wedge 0 \leq x_0$	CONJUNCIONINTRODUCCION$(1,3)$
5.	$\forall x_1 \forall x_2\ ((x_1 \leq x_2 \wedge x_2 \leq x_1) \rightarrow x_1 \equiv x_2)$	AXIOMAPROPIO
6.	$\forall x_2\ ((x_0 \leq x_2 \wedge x_2 \leq x_0) \rightarrow x_0 \equiv x_2)$	PARTICULARIZACION(5)
7.	$((x_0 \leq 0 \wedge 0 \leq x_0) \rightarrow x_0 \equiv 0)$	PARTICULARIZACION(6)
8.	$x_0 \equiv 0$	TESIS1MODUSPONENS$(4,7)$
9.	$x_0 \leq 0 \rightarrow x_0 \equiv 0$	CONCLUSION
10.	$\forall x\ (x \leq 0 \rightarrow x \equiv 0)$	GENERALIZACION(9)

(3)

1.	$x_0 + y_0 \equiv 0$	HIPOTESIS1
2.	$0 \equiv x_0 + y_0$	COMMUTATIVIDAD(1)
3.	$\exists x_3\ (0 \equiv x_0 + x_3)$	EXISTENCIAL(2)
4.	$\forall x_1 \forall x_2\ (x_1 \leq x_2 \leftrightarrow \exists x_3\ x_2 \equiv x_1 + x_3)$	AXIOMAPROPIO
5.	$x_0 \leq 0 \leftrightarrow \exists x_3\ 0 \equiv x_0 + x_3)$	PARTICULARIZACION^2(5)
6.	$x_0 \leq 0$	REEMPLAZO$(5,3)$
7.	$\forall x\ 0 \leq x$	TEOREMA
8.	$0 \leq x_0$	PARTICULARIZACION(7)
9.	$x_0 \leq 0 \wedge 0 \leq x_0$	CONJUNCIONINTRODUCCION$(6,8)$
10.	$\forall x_1 \forall x_2\ ((x_1 \leq x_2 \wedge x_2 \leq x_1) \rightarrow x_1 \equiv x_2)$	AXIOMAPROPIO
11.	$((x_0 \leq 0 \wedge 0 \leq x_0) \rightarrow x_0 \equiv 0)$	PARTICULARIZACION^2(10)
12.	$x_0 \equiv 0$	MODUSPONENS$(9,11)$
13.	$0 + y_0 \equiv 0$	REEMPLAZO$(12,1)$
14.	$\forall y\ y \equiv 0 + y$	TEOREMA
15.	$y_0 \equiv 0 + y_0$	PARTICULARIZACION(14)
16.	$y_0 \equiv 0$	TRANSITIVIDAD$(15,13)$
17.	$x_0 \equiv 0 \wedge y_0 \equiv 0$	TESIS1CONJUNCIONINTRODUCCION$(12,16)$
18.	$x_0 + y_0 \equiv 0 \rightarrow (x_0 \equiv 0 \wedge y_0 \equiv 0)$	CONCLUSION
19.	$\forall x \forall y\ (x + y \equiv 0 \rightarrow (x \equiv 0 \wedge y \equiv 0))$	GENERALIZACION^2(18)

■

Lemma 240 *Sean $n, m \in \omega$ y sea $t \in T_c^{\tau_A}$. Las siguientes sentencias son teoremas de la aritmetica de Peano:*

(a) $(+(\widehat{n}, \widehat{m}) \equiv \widehat{n+m})$

(b) $(.(\widehat{n}, \widehat{m}) \equiv \widehat{n.m})$

(c) $\forall x\ (x \leq \widehat{n} \rightarrow (x \equiv \widehat{0} \vee x \equiv \widehat{1} \vee ... \vee x \equiv \widehat{n}))$

(d) $(t \equiv \widehat{t^{\boldsymbol{\omega}}})$

Lemma 241 *Si φ es una sentencia atomica o negacion de atomica y $\boldsymbol{\omega} \models \varphi$, entonces $Arit \vdash \varphi$.*

Proof. Hay cuatro casos.

Caso $\varphi = (t \equiv s)$, con t, s terminos cerrados.

Ya que $\boldsymbol{\omega} \models \varphi$, tenemos que $t^{\boldsymbol{\omega}} = s^{\boldsymbol{\omega}}$ y por lo tanto $\widehat{t^{\boldsymbol{\omega}}} = \widehat{s^{\boldsymbol{\omega}}}$. Por el lema anterior tenemos que $Arit \vdash (t \equiv \widehat{t^{\boldsymbol{\omega}}}), (s \equiv \widehat{s^{\boldsymbol{\omega}}})$ lo cual, ya que $\widehat{t^{\boldsymbol{\omega}}}$ y $\widehat{s^{\boldsymbol{\omega}}}$ son el mismo termino nos dice por la regla de transitividad que $Arit \vdash (t \equiv s)$.

Caso $\varphi = (t \leq s)$, con t, s terminos cerrados.

Ya que $\boldsymbol{\omega} \models \varphi$, tenemos que $t^{\boldsymbol{\omega}} \leq s^{\boldsymbol{\omega}}$ y por lo tanto hay un $k \in \omega$ tal que $t^{\boldsymbol{\omega}} + k = s^{\boldsymbol{\omega}}$. Se tiene entonces que $\widehat{t^{\boldsymbol{\omega}} + k} = \widehat{s^{\boldsymbol{\omega}}}$. Por el lema anterior tenemos que $Arit \vdash +(\widehat{t^{\boldsymbol{\omega}}}, \widehat{k}) \equiv \widehat{t^{\boldsymbol{\omega}} + k}$ lo cual nos dice que

$$Arit \vdash +(\widehat{t^{\boldsymbol{\omega}}}, \widehat{k}) \equiv \widehat{s^{\boldsymbol{\omega}}}$$

Pero el lema anterior nos dice que

$$Arit \vdash (t \equiv \widehat{t^{\boldsymbol{\omega}}}), (s \equiv \widehat{s^{\boldsymbol{\omega}}})$$

y por lo tanto la regla de reemplazo nos asegura que $Arit \vdash +(t, \widehat{k}) \equiv s$. Ya que

$$\forall x_1 \forall x_2 \ (x_1 \leq x_2 \leftrightarrow \exists x_3 \ x_2 \equiv x_1 + x_3)$$

es un axioma de $Arit$, tenemos que $Arit \vdash (t \leq s)$.

Caso $\varphi = \neg(t \equiv s)$, con t, s terminos cerrados.

Caso $\varphi = \neg(t \leq s)$, con t, s terminos cerrados. ■

El siguiente lema muestra que en $Arit$ se pueden probar ciertas sentencias las cuales emulan el principio de induccion completa.

Lemma 242 *Sea $\varphi =_d \varphi(\vec{v}, v) \in F^{\tau_A}$. Supongamos v es sustituible por w en φ y $w \notin \{v_1, ..., v_n\}$. Entonces:*

$$Arit \vdash \forall \vec{v}((\varphi(\vec{v}, 0) \wedge \forall v(\forall w(w < v \rightarrow \varphi(\vec{v}, w)) \rightarrow \varphi(\vec{v}, v))) \rightarrow \forall v \varphi(\vec{v}, v))$$

Proof. Sea $\tilde{\varphi} = \forall w(w \leq v \rightarrow \varphi(\vec{v}, w))$. Notar que $Li(\tilde{\varphi}) \subseteq \{v_1, ..., v_n, v\}$. Declaremos $\tilde{\varphi} =_d \tilde{\varphi}(\vec{v}, v)$. Para hacer la prueba formal usaremos el axioma $Ind_{\tilde{\varphi}, v_1, ..., v_n, v}$. Salvo por el uso de algunos teoremas simples y el uso simultaneo de las reglas de particularizacion y generalizacion, la siguiente es la prueba formal buscada.

1.	$(\varphi(\vec{c},0) \land \forall v(\forall w(w < v \to \varphi(\vec{c},w)) \to \varphi(\vec{c},v))$	HIPOTESIS1
2.	$w_0 \leq 0$	HIPOTESIS2
3.	$\forall x\,(x \leq 0 \to x \equiv 0)$	TEOREMA
4.	$w_0 \leq 0 \to w_0 \equiv 0$	PARTICULARIZACION(3)
5.	$w_0 \equiv 0$	MODUSPONENS$(2,4)$
6.	$\varphi(\vec{c},0)$	CONJUNCIONELIMINACION(
7.	$\varphi(\vec{c},w_0)$	TESIS2REEMPLAZO$(5,6)$
8.	$w_0 \leq 0 \to \varphi(\vec{c},w_0)$	CONCLUSION
9.	$\tilde{\varphi}(\vec{c},0)$	GENERALIZACION(8)
10.	$\tilde{\varphi}(\vec{c},v_0)$	HIPOTESIS3
11.	$w_0 < v_0 + 1$	HIPOTESIS4
12.	$\forall x,y\; x < y+1 \to x \leq y$	TEOREMA
13.	$w_0 < v_0 + 1 \to w_0 \leq v_0$	PARTICULARIZACION(12)
14.	$w_0 \leq v_0$	MODUSPONENS$(11,13)$
15.	$w_0 \leq v_0 \to \varphi(\vec{c},w_0)$	PARTICULARIZACION(10)
16.	$\varphi(\vec{c},w_0)$	TESIS4MODUSPONENS$(14,15)$
17.	$w_0 < v_0 + 1 \to \varphi(\vec{c},w_0)$	CONCLUSION
18.	$\forall w\; w < v_0 + 1 \to \varphi(\vec{c},w)$	GENERALIZACION(17)
19.	$\forall v(\forall w(w < v \to \varphi(\vec{c},w)) \to \varphi(\vec{c},v))$	CONJUNCIONELIMINACION(
20.	$(\forall w(w < v_0 + 1 \to \varphi(\vec{c},w)) \to \varphi(\vec{c},v_0+1))$	PARTICULARIZACION(19)
21.	$\varphi(\vec{c},v_0+1)$	MODUSPONENS$(18,20)$
22.	$w_0 \leq v_0 + 1$	HIPOTESIS5
23.	$\forall x,y\; x \leq y+1 \to (x \leq y \lor x \equiv y+1)$	TEOREMA
24.	$w_0 \leq v_0 + 1 \to (w_0 \leq v_0 \lor w_0 \equiv v_0 + 1)$	PARTICULARIZACION(23)
25.	$(w_0 \leq v_0 \lor w_0 \equiv v_0 + 1)$	MODUSPONENS$(22,24)$
26.	$w_0 \leq v_0 \to \varphi(\vec{c},w_0)$	PARTICULARIZACION(10)
27.	$w_0 \equiv v_0 + 1$	HIPOTESIS6
28.	$\varphi(\vec{c},w_0)$	TESIS6REEMPLAZO$(21,27)$
29.	$w_0 \equiv v_0 + 1 \to \varphi(\vec{c},w_0)$	CONCLUSION
30.	$\varphi(\vec{c},w_0)$	TESIS5DISJUNCIONELIM$(25,2$
31.	$w_0 \leq v_0 + 1 \to \varphi(\vec{c},w_0)$	CONCLUSION
32.	$\tilde{\varphi}(\vec{c},v_0+1)$	TESIS3GENERALIZACION(31)
33.	$\tilde{\varphi}(\vec{c},v_0) \to \tilde{\varphi}(\vec{c},v_0+1)$	CONCLUSION
34.	$\forall v\tilde{\varphi}(\vec{c},v) \to \tilde{\varphi}(\vec{c},v+1)$	GENERALIZACION(33)
35.	$\tilde{\varphi}(\vec{c},0) \land \forall v\tilde{\varphi}(\vec{c},v) \to \tilde{\varphi}(\vec{c},v+1)$	CONJUNCIONINTRODUCCIO
36.	$Ind_{\tilde{\varphi},v_1,\ldots,v_n,v}$	AXIOMAPROPIO
37.	$(\tilde{\varphi}(\vec{c},0) \land \forall v(\tilde{\varphi}(\vec{c},v) \to \tilde{\varphi}(\vec{c},v+1)) \to \forall v\tilde{\varphi}(\vec{c},v)$	PARTICULARIZACION(36)
38.	$\forall v\tilde{\varphi}(\vec{c},v)$	MODUSPONENS$(35,37)$
39.	$\tilde{\varphi}(\vec{c},v_0)$	PARTICULARIZACION(38)
40.	$v_0 \leq v_0 \to \varphi(\vec{c},v_0)$	PARTICULARIZACION(39)
41.	$\forall x\; x \leq x$	AXIOMAPROPIO
42.	$v_0 \leq v_0$	PARTICULARIZACION(41)
43.	$\varphi(\vec{c},v_0)$	MODUSPONENS$(40,42)$
44.	$\forall v\varphi(\vec{c},v)$	TESIS1GENERALIZACION(43)
45.	$(\varphi(\vec{c},0) \land \forall v(\forall w(w < v \to \varphi(\vec{c},w)) \to \varphi(\vec{c},v))) \to \forall v\varphi(\vec{c},v)$	CONCLUSION
46.	$\forall\vec{v}((\varphi(\vec{v},0) \land \forall v(\forall w(w < v \to \varphi(\vec{v},w)) \to \varphi(\vec{v},v))) \to \forall v\varphi(\vec{v},v))$	GENERALIZACION(45)

■

6.12 Logica ecuacional

Dados $t, s \in T^\tau$, con $t \approx s$ denotaremos la siguiente sentencia de tipo τ:

$$\forall x_1 ... \forall x_n \ (t \equiv s)$$

donde n es el menor j tal que $\{x_1, ..., x_j\}$ contiene a todas las variables que ocurren en t y s. Notese que este n es 0 cuando t y s son terminos cerrados, es decir que $t \approx s$ denota a la sentencia $(t \equiv s)$, cuando $t, s \in T_c^\tau$. Las sentencias $t \approx s$, con $t, s \in T^\tau$, seran llamadas *identidades de tipo* τ. Notese que $\mathbf{A} \models t \approx s$ sii $t^{\mathbf{A}}[\vec{a}] = s^{\mathbf{A}}[\vec{a}]$, para cada $\vec{a} \in A^{\mathbf{N}}$. Tambien, si $t =_d t(x_1, ..., x_m)$ y $s =_d s(x_1, ..., x_m)$, entonces dado una τ-algebra $\mathbf{A}$, tenemos que $\mathbf{A} \models t \approx s$ sii $t^{\mathbf{A}} [a_1, ..., a_m] = s^{\mathbf{A}} [a_1, ..., a_m]$, para cada $(a_1, ..., a_m) \in A^m$. (Independientemente de que m sea el menor j tal que $\{x_1, ..., x_j\}$ contiene a las variables que ocurren en t y s.)

Una teoria de primer orden (Σ, τ) sera llamada *ecuacional* si τ es un tipo algebraico y cada elemento de Σ es una identidad de tipo τ. Por supuesto, el teorema de completitud de Godel nos garantiza que si T es una teoria ecuacional y $T \vDash t \approx s$, entonces hay una prueba formal de $t \approx s$ en T. Sin envargo, en dicha prueba formal puede haber sentencias las cuales no sean identidades. Una pregunta interesante es la siguiente:

Pregunta: ¿Hay una nocion de "prueba ecuacional" la cual sea:

- Correcta: si hay una prueba ecuacional de $t \approx s$ en T, entonces $t \approx s$ es verdadera en cada modelo de T
- Completa: si $T \vDash t \approx s$, entonces hay una prueba ecuacional de $t \approx s$ en T?

En esta seccion veremos que, tal como lo probo Birkhoff, esto es posible y que la nocion de prueba ecuacional que se puede dar es muy natural y simple, es decir si sabemos que en una teoria todos los axiomas son identidades, entonces a los fines de probar identidades las pruebas de primer orden clasicas pueden ser reemplazadas por pruebas con un formato mucho mas amigable.

6.12.1 Pruebas ecuacionales

Primero introducimos una serie de conjuntos los cuales poseen informacion deductiva ecuacional basica. Sea

$$TransEc^\tau = \{(t \approx s, s \approx p, t \approx p) : t, s, p \in T^\tau\}$$

Diremos que φ *se deduce de* ψ_1y ψ_2 *por la regla de transitividad ecuacional, respecto a* τ para expresar que $(\psi_1, \psi_2, \varphi) \in TransEc^\tau$. Sea

$$SimEc^\tau = \{(t \approx s, s \approx t) : t, s \in T^\tau\}$$

Diremos que φ *se deduce de* ψ_1 *por la regla de simetria ecuacional, respecto a* τ para expresar que $(\psi_1, \varphi) \in SimEc^\tau$. Sea

$$SubsEc^\tau = \{(t \approx s, t(p_1, ..., p_n) \approx s(p_1, ..., p_n)) : t =_d t(x_1, ..., x_n)$$
$$s =_d s(x_1, ..., x_n) \text{ y } p_1, ..., p_n \in T^\tau\}$$

Diremos que φ *se deduce de* ψ_1 *por la regla de substitucion ecuacional, respecto a* τ para expresar que $(\psi_1, \varphi) \in SubsEc^\tau$. Sea

$$ReempEc^\tau = \{(t \approx s, r \approx \tilde{r}) : t, s, r \in T^\tau \text{ y } \tilde{r} = \text{resultado}$$
$$\text{de reemplazar algunas ocurrencias de } t \text{ en } r \text{ por } s\}$$

Diremos que φ *se deduce de* ψ_1 *por la regla de reemplazo ecuacional, respecto a* τ para expresar que $(\psi_1, \varphi) \in ReempEc^\tau$.

La identidad $x_1 \approx x_1$ sera llamada *axioma logico ecuacional de tipo* τ. Notese que dicha identidad no es ni mas ni menos que la sentencia $\forall x_1(x_1 \equiv x_1)$ la cual es universalmente valida.

Definicion de prueba ecuacional Dada una teoria ecuacional (Σ, τ) y una identidad $p \approx q$ de tipo τ, una *prueba ecuacional* de $p \approx q$ en (Σ, τ) sera una palabra $\boldsymbol{\varphi} \in S^{\tau+}$ tal que

(1) Cada $\boldsymbol{\varphi}_k$, con $k = 1, ..., n(\boldsymbol{\varphi})$, es una identidad de tipo τ y $\boldsymbol{\varphi}_{n(\boldsymbol{\varphi})} = p \approx q$

(2) Para cada $k = 1, ..., n(\boldsymbol{\varphi})$, se da alguna de las siguientes

  (a) $\boldsymbol{\varphi}_k = x_1 \approx x_1$
  (b) $\boldsymbol{\varphi}_k \in \Sigma$
  (c) hay $i, j < k$ tales que $\boldsymbol{\varphi}_k$ se deduce por la regla de transitividad ecuacional a partir de $\boldsymbol{\varphi}_i$ y $\boldsymbol{\varphi}_j$
  (d) hay $i < k$ tal que $\boldsymbol{\varphi}_k$ se deduce por la regla de simetria ecuacional a partir de $\boldsymbol{\varphi}_i$
  (e) hay $i < k$ tal que $\boldsymbol{\varphi}_k$ se deduce por la regla de substitucion ecuacional a partir de $\boldsymbol{\varphi}_i$
  (f) hay $i < k$ tal que $\boldsymbol{\varphi}_k$ se deduce por la regla de reemplazo ecuacional a partir de $\boldsymbol{\varphi}_i$

Escribiremos $(\Sigma, \tau) \vdash_{ec} p \approx q$ cuando haya una prueba ecuacional de $p \approx q$ en (Σ, τ).

6.12.2 Correccion ecuacional

Para probar que el concepto de prueba ecuacional es correcto nos hara falta el siguiente lema.

Lemma 243 *Todas las reglas introducidas en la seccion anterior son universales en el sentido que si φ se deduce de $\psi_1, ..., \psi_k$ por alguna de estas reglas, entonces $((\psi_1 \wedge ... \wedge \psi_k) \rightarrow \varphi)$ es una sentencia universalmente valida.*

Proof. Veamos que la regla de reemplazo es universal. Basta con ver por induccion en k que

- Teo_k: Sean $t, s \in T^\tau$, $r \in T_k^\tau$ y sea $\mathbf{A}$ una τ-algebra tal que $t^{\mathbf{A}}[\vec{a}] = s^{\mathbf{A}}[\vec{a}]$, para cada $\vec{a} \in A^{\mathbf{N}}$. Entonces $r^{\mathbf{A}}[\vec{a}] = \tilde{r}^{\mathbf{A}}[\vec{a}]$, para cada $\vec{a} \in A^{\mathbf{N}}$, donde $\tilde{r}$ es el resultado de reemplazar algunas ocurrencias de t en r por s.

La prueba de Teo_0 es dejada al lector. Asumamos que vale Teo_k y probemos que vale Teo_{k+1}. Sean $t, s \in T^\tau$, $r \in T_{k+1}^\tau - T_k^\tau$ y sea $\mathbf{A}$ una τ-algebra tal que $t^{\mathbf{A}}[\vec{a}] = s^{\mathbf{A}}[\vec{a}]$, para cada $\vec{a} \in A^{\mathbf{N}}$. Sea $\tilde{r}$ el resultado de reemplazar algunas ocurrencias de t en r por s. El caso $t = r$ es trivial. Supongamos entonces que $t \neq r$. Supongamos $r = f(r_1, ..., r_n)$, con $r_1, ..., r_n \in T_k^\tau$ y $f \in \mathcal{F}_n$. Notese que por Lema 181 tenemos que $\tilde{r} = f(\tilde{r}_1, ..., \tilde{r}_n)$, donde cada $\tilde{r}_i$ es el resultado de reemplazar algunas ocurrencias de t en r_i por s. Para $\vec{a} \in A^{\mathbf{N}}$ se tiene que

$$\begin{array}{rcll} r^{\mathbf{A}}[\vec{a}] & = & f(r_1, ..., r_n)^{\mathbf{A}}[\vec{a}] & \\ & = & f^{\mathbf{A}}(r_1^{\mathbf{A}}[\vec{a}], ..., r_n^{\mathbf{A}}[\vec{a}]) & \\ & = & f^{\mathbf{A}}(\tilde{r}_1^{\mathbf{A}}[\vec{a}], ..., \tilde{r}_n^{\mathbf{A}}[\vec{a}]) & \text{por Teo}_k \\ & = & f(\tilde{r}_1, ..., \tilde{r}_n)^{\mathbf{A}}[\vec{a}] & \\ & = & \tilde{r}^{\mathbf{A}}[\vec{a}] & \end{array}$$

lo cual prueba Teo_{k+1}

Veamos que la regla de substitucion es universal. Supongamos $\mathbf{A} \models t \approx s$, con $t =_d t(x_1, ..., x_n)$ y $s =_d s(x_1, ..., x_n)$. Veremos que entonces $\mathbf{A} \models t(p_1, ..., p_n) \approx s(p_1, ..., p_n)$. Supongamos que $p_i =_d p_i(x_1, ..., x_m)$, para cada $i = 1, ..., n$. Por (a) del Lema 206, tenemos que

$$\begin{array}{rcl} t(p_1, ..., p_n) & = & _d t(p_1, ..., p_n)(x_1, ..., x_m) \\ s(p_1, ..., p_n) & = & _d s(p_1, ..., p_n)(x_1, ..., x_m) \end{array}$$

Sea $\vec{a} \in A^m$. Tenemos que

$$\begin{array}{rcl} t(p_1, ..., p_n)^{\mathbf{A}}[\vec{a}] & = & t^{\mathbf{A}}\left[p_1^{\mathbf{A}}[\vec{a}], ..., p_n^{\mathbf{A}}[\vec{a}]\right] \\ & = & s^{\mathbf{A}}\left[p_1^{\mathbf{A}}[\vec{a}], ..., p_n^{\mathbf{A}}[\vec{a}]\right] \\ & = & s(p_1, ..., p_n)^{\mathbf{A}}[\vec{a}] \end{array}$$

lo cual nos dice que $\mathbf{A} \models t(p_1, ..., p_n) \approx s(p_1, ..., p_n)$ ■

Theorem 244 (Correccion) *Si $(\Sigma, \tau) \vdash_{ec} p \approx q$, entonces $(\Sigma, \tau) \models p \approx q$.*

Proof. Sea $\boldsymbol{\varphi}$ una prueba ecuacional de $p \approx q$ en (Σ, τ). Usando el lema anterior se puede probar facilmente por induccion en i que $(\Sigma, \tau) \models \boldsymbol{\varphi}_i$, por lo cual $(\Sigma, \tau) \models p \approx q$ ■

6.12.3 Completitud ecuacional

Para probar que el concepto de prueba ecuacional es completo nos haran falta algunos resultados basicos que tienen interes por si mismos.

El algebra de terminos Dado un tipo algebraico τ, hay una forma natural de definir un algebra $\mathbf{T}^\tau$ cuyo universo es T^τ, de la siguiente manera

(1) $c^{\mathbf{T}^\tau} = c$, para cada $c \in \mathcal{C}$

(2) $f^{\mathbf{T}^\tau}(t_1, ..., t_n) = f(t_1, ..., t_n)$, para todo $t_1, ..., t_n \in T^\tau$, $f \in \mathcal{F}_n$.

Llamaremos a $\mathbf{T}^\tau$ el *algebra de terminos de tipo* τ.

Example 245 *Supongamos* $\tau = (\emptyset, \{f\}, \emptyset, \{(f, 1)\})$. *Entonces el universo de* $\mathbf{T}^\tau$ *es*

$$\begin{array}{l} \{x_1, f(x_1), f(f(x_1)), ...\} \cup \\ \{x_2, f(x_2), f(f(x_2)), ...\} \cup \\ \{x_3, f(x_3), f(f(x_3)), ...\} \cup \\ \qquad \vdots \end{array}$$

La funcion que interpreta a f *en* $\mathbf{T}^\tau$ *es la que a cada elemento del conjunto anterior le asigna el primer elemento que esta a su derecha. Notese entonces que* $\mathbf{T}^\tau$ *resulta isomorfa al algebra* $\mathbf{A}$ *definida por*

$$\begin{array}{rcl} A & = & \mathbf{N} \times \mathbf{N} \\ f^{\mathbf{A}}((n, m)) & = & (n, m+1) \end{array}$$

Lemma 246 *Dados* $t_1, ..., t_n$, $t =_d t(x_1, ..., x_n) \in T^\tau$, *se tiene que* $t^{\mathbf{T}^\tau}[t_1, ..., t_n] = t(t_1, ..., t_n)$.

Proof. Para cada $k \geq 0$, sea

- Teo$_k$: Dados $t_1, ..., t_n \in T^\tau$ y $t =_d t(x_1, ..., x_n) \in T_k^\tau$, se tiene que $t^{\mathbf{T}^\tau}[t_1, ..., t_n] = t(t_1, ..., t_n)$.

Veamos que es cierto Teo$_0$. Hay dos casos

Caso $t =_d t(x_1, ..., x_n) = c \in \mathcal{C}$.

Entonces tenemos

$$\begin{array}{rcl} t^{\mathbf{T}^\tau}[t_1, ..., t_n] & = & c^{\mathbf{T}^\tau} \\ & = & c \\ & = & t(t_1, ..., t_n) \end{array}$$

Caso $t =_d t(x_1, ..., x_n) = x_i$, para algun i.

Entonces tenemos

$$\begin{array}{rcl} t^{\mathbf{T}^\tau}[t_1,...,t_n] & = & t_i \\ & = & t(t_1,...,t_n) \end{array}$$

Veamos que Teo$_k$ implica Teo$_{k+1}$. Supongamos que vale Teo$_k$. Sean $t_1,...,t_n \in T^\tau$ y $t =_d t(x_1,...,x_n) \in T^\tau_{k+1} - T^\tau_k$. Hay $f \in \mathcal{F}_m$, con $m \geq 1$, y terminos $s_1,...,s_m \in T^\tau_k$ tales que $t = f(s_1,...,s_m)$. Notese que $s_i =_d s_i(x_1,...,x_n)$, $i = 1,...,m$. Tenemos entonces que

$$\begin{array}{rcl} t^{\mathbf{T}^\tau}[t_1,...,t_n] & = & f(s_1,...,s_m)^{\mathbf{T}^\tau}[t_1,...,t_n] \\ & = & f^{\mathbf{T}^\tau}(s_1^{\mathbf{T}^\tau}[t_1,...,t_n],...,s_m^{\mathbf{T}^\tau}[t_1,...,t_n]) \\ & = & f^{\mathbf{T}^\tau}(s_1(t_1,...,t_n),...,s_m(t_1,...,t_n)) \\ & = & f(s_1(t_1,...,t_n),...,s_m(t_1,...,t_n)) \\ & = & t(t_1,...,t_n) \end{array}$$

con lo cual vale Teo$_{k+1}$ ■

El algebra de terminos tiene la siguiente propiedad fundamental:

Lemma 247 (Universal Maping Property) *Si* $\mathbf{A}$ *es cualquier* τ*-algebra y* $F : Var \to A$*, es una funcion cualquiera, entonces* F *puede ser extendida a un homomorfismo* $\bar{F} : \mathbf{T}^\tau \to \mathbf{A}$.

Proof. Definamos $\bar{F}$ de la siguiente manera:

$$\bar{F}(t) = t^{\mathbf{A}}[(F(x_1), F(x_2), ...)]$$

Es claro que $\bar{F}$ extiende a F. Veamos que es un homomorfismo. Dada $c \in \mathcal{C}$, tenemos que

$$\begin{array}{rcl} \bar{F}(c^{\mathbf{T}^\tau}) & = & \bar{F}(c) \\ & = & c^{\mathbf{A}}[(F(x_1), F(x_2), ...)] \\ & = & c^{\mathbf{A}} \end{array}$$

Dados $f \in \mathcal{F}_n$, $t_1,...,t_n \in T^\tau$ tenemos que

$$\begin{array}{rcl} \bar{F}(f^{\mathbf{T}^\tau}(t_1,...,t_n)) & = & \bar{F}(f(t_1,...,t_n)) \\ & = & f(t_1,...,t_n)^{\mathbf{A}}[(F(x_1), F(x_2), ...)] \\ & = & f^{\mathbf{A}}(t_1^{\mathbf{A}}[(F(x_1), F(x_2), ...)],...,t_n^{\mathbf{A}}[(F(x_1), F(x_2), ...)]) \\ & = & f^{\mathbf{A}}(\bar{F}(t_1),...,\bar{F}(t_n)) \end{array}$$

con lo cual hemos probado que $\bar{F}$ es un homomorfismo ■

Theorem 248 (Completitud ecuacional) *Sea* (Σ, τ) *una teoria ecuacional. Si* $(\Sigma, \tau) \models p \approx q$*, entonces* $(\Sigma, \tau) \vdash_{ec} p \approx q$.

Proof. Supongamos $(\Sigma, \tau) \models p \approx q$. Sea θ la siguiente relacion binaria sobre T^τ:

$$\theta = \{(t, s) : (\Sigma, \tau) \vdash_{ec} t \approx s\}.$$

Dejamos al lector probar que θ es una congruencia de $\mathbf{T}^\tau$. Veamos que

(*) $t^{\mathbf{T}^\tau/\theta}[t_1/\theta,...,t_n/\theta]=t(t_1,...,t_n)/\theta$, para todo $t_1,...,t_n$, $t=_d t(x_1,...,x_n)$

Por Corolario 200 tenemos que

$$t^{\mathbf{T}^\tau/\theta}[t_1/\theta,...,t_n/\theta]=t^{\mathbf{T}^\tau}[t_1,...,t_n]/\theta$$

Pero por Lema 246 tenemos que $t^{\mathbf{T}^\tau}[t_1,...,t_n]=t(t_1,...,t_n)$ por lo cual (*) es verdadera.

Veamos que $\mathbf{T}^\tau/\theta\models\Sigma$. Sea $t\approx s$ un elemento de Σ, con $t=_d t(x_1,...,x_n)$ y $s=_d s(x_1,...,x_n)$. Veremos que $\mathbf{T}^\tau/\theta\models t\approx s$, es decir veremos que

$$t^{\mathbf{T}^\tau/\theta}[t_1/\theta,...,t_n/\theta]=s^{\mathbf{T}^\tau/\theta}[t_1/\theta,...,t_n/\theta]$$

para todo $t_1/\theta,...,t_n/\theta\in T^\tau/\theta$. Notese que

$$(\Sigma,\tau)\vdash_{ec} t(t_1,...,t_n)\approx s(t_1,...,t_n)$$

por lo cual $t(t_1,...,t_n)/\theta=s(t_1,...,t_n)/\theta$. Por (*) tenemos entonces

$$t^{\mathbf{T}^\tau/\theta}[t_1/\theta,...,t_n/\theta]=t(t_1,...,t_n)/\theta=s(t_1,...,t_n)/\theta=s^{\mathbf{T}^\tau/\theta}[t_1/\theta,...,t_n/\theta],$$

lo cual nos dice que $\mathbf{T}^\tau/\theta$ satisface la identidad $t\approx s$.

Ya que $\mathbf{T}^\tau/\theta\models\Sigma$, por hipotesis tenemos que $\mathbf{T}^\tau/\theta\models p\approx q$. Es decir que si $p=_d p(x_1,...,x_n)$ y $q=_d q(x_1,...,x_n)$ tenemos que $p^{\mathbf{T}^\tau/\theta}[t_1/\theta,...,t_n/\theta]=q^{\mathbf{T}^\tau/\theta}[t_1/\theta,...,t_n/\theta]$, para todo $t_1,...,t_n\in T^\tau$. En particular, tomando $t_i=x_i$, $i=1,...,n$ tenemos que

$$p^{\mathbf{T}^\tau/\theta}[x_1/\theta,...,x_n/\theta]=q^{\mathbf{T}^\tau/\theta}[x_1/\theta,...,x_n/\theta]$$

lo cual por (*) nos dice que $p/\theta=q/\theta$, produciendo $(\Sigma,\tau)\vdash_{ec} p\approx q$. ■

Corollary 249 *Sea* (Σ,τ) *una teoria ecuacional. Si* $(\Sigma,\tau)\vdash p\approx q$*, entonces* $(\Sigma,\tau)\vdash_{ec} p\approx q$.

@@finpagina@@

6.13 Teorema de incompletitud

Sea

$$Verd_{\boldsymbol{\omega}}=\{\varphi\in S^{\tau_A}:\boldsymbol{\omega}\models\varphi\}.$$

Notese que por el teorema de correccion tenemos que todo teorema de *Arit* pertenece a $Verd_{\boldsymbol{\omega}}$. Como puede notarse a medida que uno se va familiarizando con la teoria *Arit*, todos los resultados clasicos de la aritmetica los cuales pueden ser enunciados por medio de una sentencia de $Verd_{\boldsymbol{\omega}}$ son en realidad teoremas de *Arit*. Sin envargo Godel probo en su famoso teorema de incompletitud (1931) que hay una sentencia de $Verd_{\boldsymbol{\omega}}$ la cual no es un teorema de *Arit*. Por años nadie fue capaz de dar una sentencia de $Verd_{\boldsymbol{\omega}}$ la cual tenga un genuino interes aritmetico y la cual no sea un teorema de *Arit*. Recien en 1977 Paris y Harrington dieron el primer ejemplo de una tal sentencia. Una ves sabido que los axiomas de *Arit* no son suficientemente poderosos como para probar toda sentencia verdadera en $\boldsymbol{\omega}$, una pregunta interesante es

- Hay un conjunto "razonable" de axiomas $\Gamma \subseteq Verd_{\boldsymbol{\omega}}$ tal que toda sentencia de $Verd_{\boldsymbol{\omega}}$ es un teorema de (Γ, τ_A)

Una respuesta negativa a este problema tambien es dada por el teorema de incompletitud de Godel. En esta seccion daremos una prueba basada en las ideas de la computabilidad.

6.13.1 Analisis de recursividad del lenguaje de primer orden

En esta seccion estudiaremos la recursividad de la sintaxis de τ_A. Los resultados obtenidos valen para un tipo cualquiera y hemos elejido a τ_A solo para facilitar la exposicion.

Analizaremos la recursividad del concepto de prueba formal en una teoria de la forma (Σ, τ_A), donde Σ es un conjunto recursivamente enumerable. Para hacer mas concreto el tratamiento supondremos que los nombres de constante auxiliares en las pruebas formales estaran siempre en el conjunto

$$Aux = \{\triangle\square\triangle, \triangle\square\square\triangle, \triangle\square\square\square\triangle, ...\}$$

Esto no afectara nuestro analisis ya que es claro que toda prueba formal de una teoria de la forma (Σ, τ_A) puede ser reemplazada por una que sus nombres de constante auxiliares esten en Aux. Es decir que las sentencias involucradas en las pruebas formales que consideraremos seran sentencias de tipo τ_A^e donde

$$\tau_A^e = (\{0,1\} \cup Aux, \{+^2, .^2\}, \{\leq^2\}, a)$$

Sea $\mathcal{A}$ el alfabeto formado por los siguientes simbolos

$\forall\ \exists\ \neg\ \vee\ \wedge\ \rightarrow\ \leftrightarrow\ (\ )\ ,\ \equiv\ 0\ 1\ +\ .\ \leq\ \triangle\ \square$ X *0 1 ... 9* **0 1 ... 9**

Notese que los simbolos del alfabeto $\mathcal{A}$ son justamente los simbolos que ocurren en las formulas de tipo τ_A^e.

Lemma 250 *Los conjuntos $T^{\tau_A^e}$, $F^{\tau_A^e}$, T^{τ_A} y F^{τ_A} son $\mathcal{A}$-recursivos.*

Proof. Notese que los conjuntos $T^{\tau_A^e}$, $F^{\tau_A^e}$, T^{τ_A} y F^{τ_A} son $\mathcal{A}$-efectivamente computables (justifique). Entonces la Tesis de Church nos garantiza que dichos conjuntos son $\mathcal{A}$-recursivos.

A continuacion daremos una prueba de que dichos conjuntos son en realidad $\mathcal{A}$-primitivos recursivos. Veamos por ejemplo que $T^{\tau_A^e}$ es $\mathcal{A}$-primitivo recursivo. Fijemos un orden total $\leq$ sobre $\mathcal{A}$. Sea $P = \lambda x[*^{\leq}(x) \in T^{\tau_A^e}]$. Notese que $P(0) = 0$ y $P(x+1) = 1$ si y solo si se da alguna de las siguientes

- $*^{\leq}(x+1) \in \{0,1\} \cup Aux$
- $(\exists u, v \in \omega)\, *^{\leq}(x+1) = +(*^{\leq}(u), *^{\leq}(v)) \wedge (P^{\downarrow}(x))_{u+1} \wedge (P^{\downarrow}(x))_{v+1}$
- $(\exists u, v \in \omega)\, *^{\leq}(x+1) = .(*^{\leq}(u), *^{\leq}(v)) \wedge (P^{\downarrow}(x))_{u+1} \wedge (P^{\downarrow}(x))_{v+1}$

Por el Lema 80 tenemos que P es $\mathcal{A}$-p.r., por lo cual $\chi^{\mathcal{A}^*}_{T^{\tau^e_A}} = P \circ \#^{\leq}$ lo es. Notese que

$$t \in T^{\tau_A} \text{ sii } t \in T^{\tau^e_A} \wedge \triangle \text{ no ocurre en } t \wedge \square \text{ no ocurre en } t$$

por lo cual T^{τ_A} es $\mathcal{A}$-p.r. ■

Recordemos que en la Subseccion 6.7.3 definimos cuando "*v ocurre libremente en φ a partir de i*", para el caso en que $v \in Var$, $\varphi \in F^\tau$ y $i \in \{1, ..., |\varphi|\}$. Extendamos esta definicion diciendo que cuando $v \in Var$, $\varphi \in F^\tau$ y $i \in \omega - \{1, ..., |\varphi|\}$, se da que *$v$ no ocurre libremente en φ a partir de i*.

Lemma 251 *Los siguientes predicados son $\mathcal{A}$-r.*

(1) "v ocurre libremente en φ a partir de i" $: \omega \times Var \times F^{\tau^e_A} \to \omega$

(2) "$v \in Li(\varphi)$" $: Var \times F^{\tau^e_A} \to \omega$

(3) "v es sustituible por t en φ" $: Var \times T^{\tau^e_A} \times F^{\tau^e_A} \to \omega$

Proof. Notese que los predicados dados en (1), (2) y (3) son $\mathcal{A}$-efectivamente computables (justifique). Entonces la Tesis de Church nos garantiza que dichos predicados son $\mathcal{A}$-recursivos.

En realidad dichos predicados son $\mathcal{A}$-p.r.. Veamos por ejemplo que $P : \omega \times Var \times F^{\tau^e_A} \to \omega$, dado por

$$P(i, v, \varphi) = \begin{cases} 1 & \text{si } v \text{ ocurre libremente en } \varphi \text{ a partir de } i \\ 0 & \text{caso contrario} \end{cases}$$

es $\mathcal{A}$-p.r.. Sea $R : \mathbf{N} \times Var \to \omega$ el predicado dado por $R(x, v) = 1$ si y solo si $*^{\leq}((x)_1) \in F^{\tau^e_A}$ y v ocurre libremente en $*^{\leq}((x)_1)$ a partir de $(x)_2$. Sea $\bar{R} = R \cup C^{1,1}_0|_{\{0\}\times Var}$. Nex $= \{\wedge, \vee, \to, \leftrightarrow\}$. Notese que $F^{\tau^e_A}_0$ es $\mathcal{A}$-p.r. ya que

$$F^{\tau^e_A}_0 = F^{\tau^e_A} \cap (\mathcal{A} - \{\forall, \exists, \neg, \vee, \wedge, \to, \leftrightarrow\})^*$$

Notese que $\bar{R}(0, v) = 0$, para cada $v \in Var$ y que $\bar{R}(x+1, v) = 1$ si y solo si $(x+1)_2 \geq 1$ y se da alguna de las siguientes:

- $*^{\leq}((x+1)_1) \in F^{\tau^e_A}_0 \wedge v$ ocurre en $*^{\leq}((x+1)_1)$ a partir de $(x+1)_2$
- $(\exists \varphi_1, \varphi_2 \in F^{\tau^e_A})(\exists \eta \in \text{Nex}) *^{\leq}((x+1)_1) = (\varphi_1 \eta \varphi_2) \wedge$
  $\left((\bar{R}^{\downarrow}(x, v))_{\langle \#^{\leq}(\varphi_1), (x+1)_2 - 1\rangle + 1} = 1 \vee (\bar{R}^{\downarrow}(x, v))_{\langle \#^{\leq}(\varphi_2), (x+1)_2 - |(\varphi_1 \eta|\rangle + 1} = 1\right)$
- $(\exists \varphi_1 \in F^{\tau^e_A}) *^{\leq}((x+1)_1) = \neg\varphi_1 \wedge (\bar{R}^{\downarrow}(x, v))_{\langle \#^{\leq}(\varphi_1), (x+1)_2 - 1\rangle + 1} = 1$
- $(\exists \varphi_1 \in F^{\tau^e_A})(\exists w \in Var)(Q \in \{\forall, \exists\})\ w \neq v \wedge$
  $*^{\leq}((x+1)_1) = Qw\varphi_1 \wedge (\bar{R}^{\downarrow}(x, v))_{\langle \#^{\leq}(\varphi_1), (x+1)_2 - |(Qw|\rangle + 1} = 1$

Es decir que por el Lema 80 tenemos que $\bar{R}$ es $\mathcal{A}$-p.r.. Notese que para $(i,v,\varphi)\in\omega\times Var\times F^{\tau_A^e}$, tenemos $P(i,v,\varphi)=\bar{R}(\langle\#^{\leq}(\varphi),i\rangle,v)$. Ahora es facil obtener la funcion P haciendo composiciones adecuadas con $\bar{R}$. ■

Dados $v\in Var$ y $t,s\in T^{\tau_A^e}$, usaremos $\downarrow_v^t(s)$ para denotar el resultado de reemplazar simultaneamente cada ocurrencia de v en s por t. Similarmente, si $\varphi\in F^{\tau_A^e}$, usaremos $\downarrow_v^t(\varphi)$ para denotar el resultado de reemplazar simultaneamente cada ocurrencia libre de v en φ por t.

Lemma 252 *Las funciones $\lambda svt[\downarrow_v^t(s)]$ y $\lambda\varphi vt[\downarrow_v^t(\varphi)]$ son $\mathcal{A}$-r.*

Proof. Notese que las funciones $\lambda svt[\downarrow_v^t(s)]$ y $\lambda\varphi vt[\downarrow_v^t(\varphi)]$ son $\mathcal{A}$-efectivamente computables (justifique). Entonces la Tesis de Church nos garantiza que dichas funciones son $\mathcal{A}$-recursivas.

En realidad son $\mathcal{A}$-p.r.. Veamos por ejemplo que $\lambda svt[\downarrow_v^t(s)]$ es $\mathcal{A}$-p.r.. Sea $\leq$ un orden total sobre $\mathcal{A}$. Sea $h:\omega\times Var\times T^{\tau_A^e}\rightarrow\omega$ dada por

$$h(x,v,t)=\begin{cases}\#^{\leq}(\downarrow_v^t(*^{\leq}(x))) & \text{si } *^{\leq}(x)\in T^{\tau_A^e}\\ 0 & \text{caso contrario}\end{cases}$$

Sea $P:\mathbf{N}\times\omega\times Var\times T^{\tau_A^e}\times\mathcal{A}^*\rightarrow\omega$ tal que $P(A,x,v,t,\alpha)=1$ si y solo si se da alguna de las siguientes

- $*^{\leq}(x+1)\notin T^{\tau_A^e}\wedge\alpha=\varepsilon$
- $*^{\leq}(x+1)=v\wedge\alpha=t$
- $*^{\leq}(x+1)\in(\{0,1\}\cup Aux)-\{v\}\wedge\alpha=*^{\leq}(x+1)$
- $(\exists r,s\in T^{\tau_A^e})*^{\leq}(x+1)=+(r,s)\wedge\alpha=+(*^{\leq}((A)_{\#^{\leq}(r)+1}),*^{\leq}((A)_{\#^{\leq}(s)+1}))$
- $(\exists r,s\in T^{\tau_A^e})*^{\leq}(x+1)=.(r,s)\wedge\alpha=.(*^{\leq}((A)_{\#^{\leq}(r)+1}),*^{\leq}((A)_{\#^{\leq}(s)+1}))$

Sea $\bar{P}=P\cup C_0^{2,2}|_{\{0\}\times\omega\times Var\times T^{\tau_A^e}\times\mathcal{A}^*}$. Notese que $\bar{P}(h^{\downarrow}(x,v,t),x,v,t,\alpha)=1$ si y solo si ya sea $*^{\leq}(x+1)\notin T^{\tau_A^e}$ y $\alpha=\varepsilon$ o $*^{\leq}(x+1)\in T^{\tau_A^e}$ y $\alpha=\downarrow_v^t(*^{\leq}(x+1))$. Tenemos entonces

$$\begin{array}{rcl}h(0,v,t)&=&0\\ h(x+1,v,t)&=&\#^{\leq}(\min\limits_{\alpha}^{\leq}\bar{P}(h^{\downarrow}(x,v,t),x,v,t,\alpha)),\end{array}$$

por lo cual el Lema 80 nos dice que h es $\mathcal{A}$-p.r. Ahora es facil obtener la funcion $\lambda svt[\downarrow_v^t(s)]:T^{\tau_A^e}\times Var\times T^{\tau_A^e}\rightarrow T^{\tau_A^e}$ haciendo composiciones adecuadas con h. ■

Lemma 253 *El predicado $R:\mathcal{A}^4\rightarrow\omega$, dado por*

$$R(\alpha,\beta,\gamma,\zeta)=\left\{\begin{array}{lll} 1 & si\ \beta= & resultado\ de\ reemplazar\ una \\ & & ocurrencia\ de\ \gamma\ en\ \alpha\ por\ \zeta \\ 0 & caso\ contrario & \end{array}\right.$$

es $\mathcal{A}$-r..

Proof. Notese que el predicado R es $\mathcal{A}$-efectivamente computable. Entonces la Tesis de Church nos garantiza que R es $\mathcal{A}$-recursivo.

En realidad R es $\mathcal{A}$-p.r. y esto puede verse facilmente ya que $R(\alpha,\beta,\gamma,\zeta)=1$ sii existen α_1,α_2 tales que $\alpha=\alpha_1\gamma\alpha_2$ y $\beta=\alpha_1\zeta\alpha_2$. ■

Lemma 254 *Los conjuntos $ModPon^{\tau_A^e}$, $Elec^{\tau_A^e}$, $Reem^{\tau_A^e}$, $ConjInt^{\tau_A^e}$, $ConjElim^{\tau_A^e}$, $EquivInt^{\tau_A^e}$, $DisjElim^{\tau_A^e}$, $DisjInt^{\tau_A^e}$, $EquivElim^{\tau_A^e}$, $Generaliz^{\tau_A^e}$, $Commut^{\tau_A^e}$, $Trans^{\tau_A^e}$, $Exist^{\tau_A^e}$, $Evoc^{\tau_A^e}$, $Absur^{\tau_A^e}$, $DivPorCas^{\tau_A^e}$, $Partic^{\tau_A^e}$ son $\mathcal{A}$-r..*

Proof. Dejamos al lector una prueba via la Tesis de Church. En realidad dichos conjuntos son $\mathcal{A}$-p.r.. Veremos, por ejemplo que $Reem^{\tau_A^e}$ es $\mathcal{A}$-p.r.. Basta con ver que $Reem1^{\tau_A^e}$ y $Reem2^{\tau_A^e}$ lo son. Veremos que $Reem2^{\tau_A^e}$ es $\mathcal{A}$-p.r.. Sea $Q:F^{\tau_A^e}\times F^{\tau_A^e}\times F^{\tau_A^e}\rightarrow\omega$ el predicado tal que $Q(\psi,\varphi,\sigma)=1$ si y solo si

$$(\exists\alpha\in(\forall Var)^*)(\exists\psi_1,\psi_2\in F^{\tau_A^e})\ \psi=\alpha(\psi_1\leftrightarrow\psi_2)\wedge$$
$$Li(\psi_1)=Li(\psi_2)\wedge((\forall v\in Var)\ v\notin Li(\psi_1)\vee v \text{ ocurre en } \alpha)$$
$$((\forall v\in Var)\ v \text{ no ocurre en } \alpha\vee v\in Li(\psi_1))\wedge R(\varphi,\sigma,\psi_1,\psi_2)$$

(R es el predicado dado por el Lema 253). Es facil ver que Q es $\mathcal{A}$-p.r. y que $\chi^{\mathcal{A}^4}_{Reem2^{\tau_A^e}}=Q|_{S^{\tau_A^e}\times S^{\tau_A^e}\times S^{\tau_A^e}}$. ■

Lemma 255 *El predicado "ψ se deduce de φ por generalizacion con constante c, con respecto a τ_A^e" : $S^{\tau_A^e}\times S^{\tau_A^e}\times Aux\rightarrow\omega$ es $\mathcal{A}$-r..*

Proof. Es claro que el predicado en cuestion es $\mathcal{A}$-efectivamente computable (justifique). Por la Tesis de Church tenemos entonces que dicho predicado es $\mathcal{A}$-r.

Para probar que en realidad dicho predicado es $\mathcal{A}$-p.r., notese que: ψ se deduce de φ por generalizacion con constante c si y solo si hay una formula γ y una variable v tales que

- $Li(\gamma)=\{v\}$
- c no ocurre en γ
- $\varphi=\downarrow_v^c(\gamma)\wedge\psi=\forall v\gamma$

■

Lemma 256 *El predicado "ψ se deduce de φ por eleccion con constante e, con respecto a τ_A^e" : $S^{\tau_A^e} \times S^{\tau_A^e} \times Aux \rightarrow \omega$ es $\mathcal{A}$-r..*

Proof. Es claro que el predicado en cuestion es $\mathcal{A}$-efectivamente computable (justifique). Por la Tesis de Church tenemos entonces que dicho predicado es $\mathcal{A}$-r.

Dejamos al lector probar que en realidad dicho predicado es $\mathcal{A}$-p.r. ■

Recordemos que

$$AxLog^{\tau_A^e} = \{\varphi \in S^{\tau_A^e} : \varphi \text{ es un axioma logico de tipo } \tau_A^e\}$$

Lemma 257 *$AxLog^{\tau_A^e}$ es $\mathcal{A}$-r..*

Proof. Es claro que el conjunto en cuestion es $\mathcal{A}$-efectivamente computable (justifique). Por la Tesis de Church tenemos entonces que dicho conjunto es $\mathcal{A}$-r.

Dejamos al lector probar que en realidad dicho conjunto es $\mathcal{A}$-p.r. La prueba es completamente analoga a la prueba de que Ins^Σ es un conjunto $(\Sigma \cup \Sigma_p)$-p.r. (Lema 95) ■

Lemma 258 *Sea Σ un alfabeto finito. Sea $S \subseteq \Sigma^*$ un conjunto Σ-r.. El conjunto S^+ es Σ-r.*

Proof. Ya que S es Σ-r., tenemos que S es Σ-efectivamente computable. Es facil ver que entonces S^+ es Σ-efectivamente computable. Por la Tesis de Church tenemos entonces que S es Σ-recursivo.

Ya que $\alpha \in S^+$ si y solo si

$$(\exists z \in \mathbf{N})(\forall i \in \mathbf{N})_{i \leq Lt(z)} *^{\leq} ((z)_i) \in S \wedge \alpha = \subset_{i=1}^{Lt(z)} *^{\leq} ((z)_i)$$

se puede probar tambien este lema sin usar la Tesis de Church. Dejamos al lector los detalles. ■

Recordemos que dada $\boldsymbol{\varphi} \in S^{\tau_A^e +}$, usamos $n(\boldsymbol{\varphi})$ y $\boldsymbol{\varphi}_1, ..., \boldsymbol{\varphi}_{n(\boldsymbol{\varphi})}$ para denotar los unicos n y $\varphi_1, ..., \varphi_n$ tales que $\boldsymbol{\varphi} = \varphi_1 ... \varphi_n$ (la unicidad es garantizada en Lema 215). Extendamos esta notacion definiendo $\boldsymbol{\varphi}_i = \varepsilon$ para $i = 0$ o $i > n(\boldsymbol{\varphi})$.

Lemma 259 *Las funciones*

$$\begin{array}{ccc} S^{\tau_A^e +} & \rightarrow & \omega \\ \boldsymbol{\varphi} & \rightarrow & n(\boldsymbol{\varphi}) \end{array} \qquad\qquad \begin{array}{ccc} \omega \times S^{\tau_A^e +} & \rightarrow & S^{\tau_A^e} \cup \{\varepsilon\} \\ (i, \boldsymbol{\varphi}) & \rightarrow & \boldsymbol{\varphi}_i \end{array}$$

son $\mathcal{A}$-r.

Proof. Es claro que la funciones en cuestion son $\mathcal{A}$-efectivamente computables (justifique). Por la Tesis de Church tenemos entonces que son $\mathcal{A}$-r.

Dejamos al lector probar que en realidad dicho conjunto es $\mathcal{A}$-p.r. La prueba es completamente analoga a la prueba de que $\lambda\mathcal{P}\left[n(\mathcal{P})\right]$ y $\lambda i\mathcal{P}\left[I_i^{\mathcal{P}}\right]$ son funciones $(\Sigma\cup\Sigma_p)$-p.r. (Lema 97) ■

Recordemos que dada $\mathbf{J}\in Just^+$, usamos $n(\mathbf{J})$ y $\mathbf{J}_1,...,\mathbf{J}_{n(\mathbf{J})}$ para denotar los unicos n y $J_1,...,J_n$ tales que $\mathbf{J}=J_1...J_n$ (la unicidad es garantizada en Lema 214). Extendamos esta notacion definiendo $\mathbf{J}_i=\varepsilon$ para $i=0$ o $i>n(\mathbf{J})$.

Sea $\mathcal{B}$ el alfabeto que consiste en todos los simbolos que ocurren en alguna palabra de $Just$. Es decir $\mathcal{B}$ consiste de los simbolos

( ) , 0 1 2 3 4 5 6 7 8 9 A B C D E G H I J L M N O P Q R S T U V X Z

Lemma 260 *$Just$ es $\mathcal{B}$-r. Las funciones*

$$\begin{array}{ccc} Just^+ & \rightarrow & \omega \\ \mathbf{J} & \rightarrow & n(\mathbf{J}) \end{array} \qquad\qquad \begin{array}{ccc} \omega\times Just^+ & \rightarrow & Just\cup\{\varepsilon\} \\ (i,\mathbf{J}) & \rightarrow & \mathbf{J}_i \end{array}$$

son $\mathcal{B}$-r.

Proof. Es claro que la funciones en cuestion son $\mathcal{B}$-efectivamente computables (justifique). Por la Tesis de Church tenemos entonces que son $\mathcal{B}$-r. ■

Lemma 261 *El predicado "$\langle i,j\rangle\in\mathcal{B}^{\mathbf{J}}$" $:\omega\times\omega\times Just^+\rightarrow\omega$ es $\mathcal{B}$-r*

Proof. Es claro que dicho predicado es $\mathcal{B}$-efectivamente computable (justifique). Por la Tesis de Church tenemos entonces que es $\mathcal{B}$-r. ■

Lemma 262 *El conjunto $\{\mathbf{J}\in Just^+ : \mathbf{J}$ es balanceada$\}$ es $\mathcal{B}$-r.*

Proof. Es claro que dicho conjunto es $\mathcal{B}$-efectivamente computable (justifique). Por la Tesis de Church tenemos entonces que es $\mathcal{B}$-r. ■

Lemma 263 *Los predicados*

$$\begin{array}{rcl} \omega\times S^{\tau_A^e}\times S^{\tau_A^e+}\times Just^+ & \rightarrow & \omega \\ (i,\varphi,\boldsymbol{\varphi},\mathbf{J}) & \rightarrow & \left\{\begin{array}{ll} 1 & \text{si } (\boldsymbol{\varphi},\mathbf{J}) \text{ es adecuado y } \varphi \text{ es hipotesis de } \boldsymbol{\varphi}_i \text{ en } (\boldsymbol{\varphi},\mathbf{J}) \\ 0 & \text{caso contrario} \end{array}\right. \end{array}$$

$$\begin{array}{rcl} \omega\times\omega\times S^{\tau_A^e+}\times Just^+ & \rightarrow & \omega \\ (i,\varphi,\boldsymbol{\varphi},\mathbf{J}) & \rightarrow & \left\{\begin{array}{ll} 1 & \text{si } (\boldsymbol{\varphi},\mathbf{J}) \text{ es adecuado y } i \text{ es anterior a } j \text{ en } (\boldsymbol{\varphi},\mathbf{J}) \\ 0 & \text{caso contrario} \end{array}\right. \end{array}$$

son $(\mathcal{A}\cup\mathcal{B})$-r..

Proof. Es claro que los predicados en cuestion son $(\mathcal{A}\cup\mathcal{B})$-efectivamente computables (justifique). Por la Tesis de Church tenemos entonces que dichos predicados son $(\mathcal{A}\cup\mathcal{B})$-r. ■

Lemma 264 *El predicado*

$$\begin{array}{rcl} Aux\times Aux\times S^{\tau_A^e+}\times Just^+ & \rightarrow & \omega \\ (e,d,\boldsymbol{\varphi},\mathbf{J}) & \rightarrow & \left\{\begin{array}{ll} 1 & \text{si } (\boldsymbol{\varphi},\mathbf{J}) \text{ es adecuado y } e \text{ depende de } d \text{ en } (\boldsymbol{\varphi},\mathbf{J}) \\ 0 & \text{caso contrario}\end{array}\right.\end{array}$$

es $(\mathcal{A}\cup\mathcal{B})$*-r..*

Proof. Es claro que el predicado en cuestion es $(\mathcal{A}\cup\mathcal{B})$-efectivamente computable (justifique). Por la Tesis de Church tenemos entonces que dicho predicado es $(\mathcal{A}\cup\mathcal{B})$-r. ■

Dada una teoria de la forma (Σ,τ_A), diremos que una prueba formal $(\boldsymbol{\varphi},\mathbf{J})$ de φ en (Σ,τ_A) es *normal* si solo usa nombres de ctes auxiliares de Aux, es decir si $\boldsymbol{\varphi}\in S^{\tau_A^e+}$. Definamos

$$Pruebas_{(\Sigma,\tau_A)}=\{(\boldsymbol{\varphi},\mathbf{J}):\exists\varphi\ (\boldsymbol{\varphi},\mathbf{J}) \text{ es una prueba normal de } \varphi \text{ en } (\Sigma,\tau_A)\}$$

Lemma 265 *Sea* (Σ,τ_A) *una teoria tal que* Σ *es* $\mathcal{A}$*-r.e. (resp.* $\mathcal{A}$*-recursivo). Entonces* $Pruebas_{(\Sigma,\tau_A)}$ *es* $(\mathcal{A}\cup\mathcal{B})$*-r.e. (resp.* $(\mathcal{A}\cup\mathcal{B})$*-recursivo).*

Proof. Supongamos que Σ es $\mathcal{A}$-r.e.. Claramente entonces Σ es $\mathcal{A}$-efectivamente computable por lo cual hay una funcion $g:\omega\rightarrow\Sigma$ la cual es $\mathcal{A}$-efectivamente computable y suryectiva. Sea $\leq$ un orden total sobre $\mathcal{A}\cup\mathcal{B}$. A continuacion describimos como hacer un procedimiento efectivo enumere a $Pruebas_{(\Sigma,\tau_A)}$. Dejamos al lector completar los detalles. Dada una prueba formal $(\boldsymbol{\varphi},\mathbf{J})$ cualquiera, definamos

$$Ax((\boldsymbol{\varphi},\mathbf{J}))=\{\boldsymbol{\varphi}_i:\text{existe }\alpha\text{ tal que }\mathbf{J}_i=\alpha\text{AXIOMAPROPIO}\}$$

Etapa 1
Si $x=0$, detenerse y dar como salida $((0\equiv 0),\text{AXIOMALOGICO})$. En caso contrario ir a Etapa 2

Etapa 2.
Si $(*^{\leq}((x)_1),*^{\leq}((x)_2))$ es una prueba formal y $Ax((*^{\leq}((x)_1),*^{\leq}((x)_2)))\subseteq\{g(0),...,g((x)_3)\}$, entonces detenerse y dar como salida $(*^{\leq}((x)_1),*^{\leq}((x)_2))$. Caso contrario detenerse y dar como salida $((0\equiv 0),\text{AXIOMALOGICO})$

Por la Tesis de Church tenemos entonces que $Pruebas_{(\Sigma,\tau_A)}$ es $(\mathcal{A}\cup\mathcal{B})$-r.e.
■

Dada una teoria (Σ,τ_A), definamos

$$Teo_{(\Sigma,\tau_A)}=\{\varphi\in S^{\tau_A}:(\Sigma,\tau_A)\vdash\varphi\}$$

Proposition 266 *Si* (Σ, τ_A) *es una teoria tal que* Σ *es* $\mathcal{A}$*-r.e., entonces* $Teo_{(\Sigma,\tau_A)}$ *es* $\mathcal{A}$*-r.e.*

Proof. Como se vio en el lema anterior, tenemos que $Pruebas_{(\Sigma,\tau_A)}$ es $(\mathcal{A}\cup\mathcal{B})$-efectivamente enumerable. Es facil ahora, usando un procedimiento efectivo que enumere a $Pruebas_{(\Sigma,\tau_A)}$, diseñar un procedimiento efectivo que enumere a $Teo_{(\Sigma,\tau_A)}$. Es decir que $Teo_{(\Sigma,\tau_A)}$ es $\mathcal{A}$-efectivamente enumerable, lo cual por la Tesis de Church nos dice que es $\mathcal{A}$-r.e.

A continuacion daremos una prueba que no usa la Tesis de Church. Ya que $Pruebas_{(\Sigma,\tau_A)}$ es $(\mathcal{A}\cup\mathcal{B})$-r.e. (lema anterior) tenemos que hay una funcion $F:\omega\rightarrow S^{\tau_A^e+}\times Just^+$ la cual cumple que $p_1^{0,2}\circ F$ y $p_2^{0,2}\circ F$ son $(\mathcal{A}\cup\mathcal{B})$-r. y ademas $I_F = Pruebas_{(\Sigma,\tau_A)}$. Sea

$$\begin{array}{rcl} g: S^{\tau_A^e+} & \rightarrow & S^{\tau_A^e} \\ \boldsymbol{\varphi} & \rightarrow & \varphi_{n(\boldsymbol{\varphi})} \end{array}$$

Por lemas anteriores g es $\mathcal{A}$-r.. Notese que $I_{(g\circ p_1^{0,2}\circ F)} = Teo_{(\Sigma,\tau_A)}$, lo cual dice que $Teo_{(\Sigma,\tau_A)}$ es $(\mathcal{A}\cup\mathcal{B})$-r.e. (Teorema 126). Por el teorema de independencia del alfabeto tenemos que $Teo_{(\Sigma,\tau_A)}$ es $\mathcal{A}$-r.e.. ∎

6.13.2 Funciones representables

Sea $n\geq 1$. Una funcion $f:D_f\subseteq\omega^n\rightarrow\omega$ sera llamada *representable* si hay una formula $\varphi =_d \varphi(v_1,...,v_n,v)\in F^{\tau_A}$, la cual cumpla

$$\boldsymbol{\omega}\models\varphi\left[k_1,...,k_n,k\right] \text{ si y solo si } f(k_1,...,k_n)=k$$

cualesquiera sean $k_1,...,k_n,k\in\omega$. En tal caso diremos que la formula φ *representa* a la funcion f, con respecto a la declaracion $\varphi =_d \varphi(v_1,...,v_n,v)$. Notese que cuando $(k_1,...,k_n)\notin D_f$ entonces debera suceder que $\boldsymbol{\omega}\not\models\varphi\left[k_1,...,k_n,k\right]$, cualquiera sea $k\in\omega$. Cabe destacar que una formula φ puede representar a f, con respecto a una declaracion y con respecto a otra declaracion puede no representarla. Por ejemplo la formula $(x_3\equiv x_1+x_2)$ representa a la operacion suma con respecto a las declaraciones $\varphi =_d \varphi(x_1,x_2,x_3)$ y $\varphi =_d \varphi(x_2,x_1,x_3)$ pero con respecto a la declaracion $\varphi =_d \varphi(x_3,x_2,x_1)$ no representa a dicha operacion. Para dar otro ejemplo, tomemos $\varphi = (x_5\equiv 1)$. Entonces

- Con respecto a la declaracion $\varphi =_d \varphi(x_2,x_5)$ la formula φ representa a la funcion con dominio ω y valor constantemente 1
- Con respecto a la declaracion $\varphi =_d \varphi(x_{10},x_5)$ la formula φ representa a la funcion con dominio ω y valor constantemente 1
- Con respecto a la declaracion $\varphi =_d \varphi(x_2,x_6,x_5)$ la formula φ representa a la funcion con dominio ω^2 y valor constantemente 1

El concepto de funcion representable sera clave en nuestra prueba del teorema de incompletitud. El resultado clave desde el cual sale facilmente el teorema de incompletitud es la Proposicion 272 en la que se prueba que el conjunto $Verd_{\boldsymbol{\omega}}$ no es $\mathcal{A}$-r.e..Para probar dicha proposicion primero probaremos que toda funcion $\emptyset$-recursiva es representable. Aqui es clave una funcion introducida por Godel. Sea

$$\beta = \lambda xyi[R(x, y(i+1)+1)]$$

donde

$$\begin{array}{rcl} R : \omega \times \mathbf{N} & \rightarrow & \omega \\ (x, y) & \rightarrow & \text{resto de la division de } x \text{ por } y \end{array}$$

Notese que $D_\beta = \omega^3$. Esta funcion, conocida como la *funcion β de Godel*, es representable ya que por ejemplo la formula

$$\varphi = \exists x_5 \ (x_1 \equiv x_5.(x_2.(x_3+1)+1) + x_4 \wedge x_4 < x_2.(x_3+1)+1)$$

la representa, con respecto a la declaracion $\varphi =_d \varphi(x_1, x_2, x_3, x_4)$. Ahora veremos un lema que muestra que la funcion β tiene una propiedad sorprendente en el sentido de que cualquier sucesion finita de elementos de ω es producida por β si fijamos adecuadamente sus dos primeras entradas. Dados $x, y \in \omega$, diremos que x e y son *coprimos* cuando 1 sea el unico elemento de ω que divide a ambos. Notese que x e y no son son coprimos sii existe un numero primo $p \in \omega$ que los divide a ambos

Lemma 267 *Cualesquiera sean $z_0, ..., z_n \in \omega$, $n \geq 0$, hay $x, y \in \omega$, tales que $\beta(x, y, i) = z_i$, $i = 0, ..., n$*

Proof. Dados $x, y, m \in \omega$ con $m \geq 1$, usaremos $x \equiv y(m)$ para expresar que x es congruente a y modulo m, es decir para expresar que $x - y$ es divisible por m. Usaremos en esta prueba el Teorema Chino del Resto:

- Dados $m_0, ..., m_n, z_0, ..., z_n \in \omega$ tales que $m_0, ..., m_n$ son coprimos de a pares, hay un $x \in \omega$ tal que $x \equiv z_i(m_i)$, para $i = 0, ..., n$.

Sea $y = \max(z_0, ..., z_n, n)!$. Sean $m_i = y(i+1)+1$, $i = 0, ..., n$. Veamos que $m_0, ..., m_n$ son coprimos de a pares. Supongamos p divide a m_i y a m_j con $i < j$. Entonces p divide a $m_j - m_i = y(j-i)$ y ya que p no puede dividir a y, tenemos que p divide a $j - i$. Pero ya que $j - i < n$ tenemos que $p < n$ lo cual es absurdo ya que implicaria que p divide y.

Por el Teorema Chino del Resto hay un x tal que $x \equiv z_i(m_i)$, para $i = 0, ..., n$. Ya que $z_i < m_i$, tenemos que

$$\beta(x, y, i) = R(x, y(i+1)+1) = R(x, m_i) = z_i, \ i = 0, ..., n.$$

■

El lema anterior nos permite probar:

Proposition 268 *Si h es $\emptyset$-recursiva, entonces h es representable*

Proof. Supongamos $f : S_1 \times ... \times S_n \rightarrow \omega$ y $g : \omega \times \omega \times S_1 \times ... \times S_n \rightarrow \omega$ son representables, con $S_1, ..., S_n \subseteq \omega$ y $n \geq 0$. Probaremos que entonces $R(f,g) : \omega \times S_1 \times ... \times S_n \rightarrow \omega$ lo es. Para esto primero notese que para $t, x_1, ..., x_n, z \in \omega$, las siguientes son equivalentes

(1) $R(f,g)(t, \vec{x}) = z$

(2) Hay $z_0, ..., z_t \in \omega$ tales que

$$\begin{array}{rcl} z_0 & = & f(\vec{x}) \\ z_{i+1} & = & g(z_i, i, \vec{x}),\ i = 0, ..., t-1 \\ z_t & = & z \end{array}$$

(3) Hay $x, y \in \omega$ tales que

$$\begin{array}{rcl} \beta(x,y,0) & = & f(\vec{x}) \\ \beta(x,y,i+1) & = & g(\beta(x,y,i), i, \vec{x}),\ i = 0, ..., t-1 \\ \beta(x,y,t) & = & z \end{array}$$

Sean φ_β, φ_f y φ_g formulas que representen a las funciones β, f y g, con respecto a las declaraciones

$$\begin{array}{rcl} \varphi_\beta & = & {}_d\varphi_\beta(x_1, x_2, x_3, x_4) \\ \varphi_f & = & {}_d\varphi_f(x_1, ..., x_n, x_{n+1}) \\ \varphi_g & = & {}_d\varphi_g(x_1, ..., x_{n+2}, x_{n+3}) \end{array}$$

respectivamente. Sean $v_1, ..., v_{n+1}, v, y_1, y_2, y_3, y_4, z_1, z_2$ variables todas distintas y tales que cada una de las variebles libres de φ_β, φ_f y φ_g es sustituible por cada una de las variables $v_1, ..., v_{n+1}, v,\ y_1, y_2, y_3, y_4, z_1, z_2$. Sea $\varphi_{R(f,g)}$ la siguiente formula

$$\exists z_1, z_2\ (\exists y_1 \varphi_\beta(z_1, z_2, 0, y_1) \wedge \varphi_f(v_2, ..., v_{n+1}, y_1)) \wedge$$
$$\varphi_\beta(z_1, z_2, v_1, v) \wedge \forall y_2 (y_2 < v_1 \rightarrow \exists y_3, y_4\ \varphi_\beta(z_1, z_2, y_2 + 1, y_3) \wedge$$
$$\varphi_\beta(z_1, z_2, y_2, y_4) \wedge \varphi_g(y_4, y_2, v_2, ..., v_{n+1}, y_3))$$

Es facil usando (3) ver que la formula $\varphi_{R(f,g)}$ representa a $R(f,g)$, con respecto a la declaracion $\varphi_{R(f,g)} =_d \varphi_{R(f,g)}(v_1, ..., v_{n+1}, v)$.

En forma analoga se puede probar que las reglas de composicion y minimizacion preservan representabilidad por lo cual ya que los elementos de $\mathrm{R}_0^\emptyset$ son representables, por induccion tenemos que lo es toda funcion $\emptyset$-r. ■

6.13.3 Prueba del teorema de incompletitud

Nuestra estrategia sera probar que $Verd_{\boldsymbol{\omega}}$ no es $\mathcal{A}$-r.e., para lo cual necesitamos los siguientes dos lemas. El primero consiste en dar una funcion total numerica la cual codifique al predicado "el programa $\mathcal{P}$ se detiene luego de t pasos, partiendo del estado $((0,0,...),(\mathcal{P},\varepsilon,...))$".

Lemma 269 *Hay un predicado $P:\omega\times\omega\rightarrow\omega$ el cual es $\emptyset$-p.r. y tal que el predicado $Q=\lambda x\left[(\exists t\in\omega)P(t,x)\right]:\omega\rightarrow\omega$ no es $\emptyset$-r.*

Proof. Sea $\Sigma=\Sigma_p$. Recordemos que el predicado

$$P_1=\lambda t\mathcal{P}\left[i^{0,1}(t,\mathcal{P},\mathcal{P})=n(\mathcal{P})+1\right]$$

es Σ_p-p.r. ya que la funcion $i^{0,1}$ lo es. Notese que el dominio de P_1 es $\omega\times\mathrm{Pro}^{\Sigma_p}$. Por Lema 133 tenemos que

$$AutoHalt^{\Sigma_p}=\lambda\mathcal{P}\left[(\exists t\in\omega)\ P_1(t,\mathcal{P})\right]$$

no es Σ_p-recursivo. Sea $\leq$ un orden total sobre Σ_p. Definamos $P:\omega\times\omega\rightarrow\omega$ de la siguiente manera

$$P(t,x)=\begin{cases} P_1(t,*^{\leq}(x)) & \text{si} \quad *^{\leq}(x)\in\mathrm{Pro}^{\Sigma_p}\\ 0 & \text{si} \quad *^{\leq}(x)\notin\mathrm{Pro}^{\Sigma_p}\end{cases}$$

Claramente P es Σ_p-p.r., por lo cual el teorema de independencia del alfabeto nos dice que P es $\emptyset$-p.r.. Sea $Q=\lambda x\left[(\exists t\in\omega)P(t,x)\right]$. Notese que

$$AutoHalt^{\Sigma_p}=Q\circ\#^{\leq}|_{\mathrm{Pro}^{\Sigma_p}}$$

lo cual dice que Q no es Σ_p-r. ya que de serlo, el predicado $AutoHalt^{\Sigma_p}$ lo seria. Por el teorema de independencia del alfabeto tenemos entonces que Q no es $\emptyset$-recursivo. ■

Corollary 270 *No toda funcion representable es $\emptyset$-recursiva*

Proof. Dejamos como ejercicio para el lector probar que el predicado Q del lema anterior es representable, lo cual completa la prueba de este corolario ya que Q no es $\emptyset$-recursivo. ■

Recordemos que para $\alpha\in\Sigma^*$, definimos

$$^{\frown}\alpha=\begin{cases}[\alpha]_2\ldots[\alpha]_{|\alpha|} & \text{si} \quad |\alpha|\geq 2\\ \varepsilon & \text{si} \quad |\alpha|\leq 1\end{cases}$$

Lemma 271 *Si $Verd_{\boldsymbol{\omega}}$ es $\mathcal{A}$-r.e., entonces es $\mathcal{A}$-r.*

Proof. Supongamos $Verd_{\boldsymbol{\omega}}$ es $\mathcal{A}$-r. e. Sea $f:\omega\rightarrow Verd_{\boldsymbol{\omega}}$ una funcion sobre y $\mathcal{A}$-r. Sea $g:S^{\tau_A}\rightarrow S^{\tau_A}$, dada por

$$g(\varphi)=\begin{cases}\frown\varphi & \text{si } [\varphi]_1=\neg\\ \neg\varphi & \text{caso contrario}\end{cases}$$

Notar que g es $\mathcal{A}$-p.r. por lo cual $g\circ f$ es $\mathcal{A}$-r. Ya que $I_{g\circ f}=S^{\tau_A}-Verd_{\boldsymbol{\omega}}$ (justifique), tenemos que $S^{\tau_A}-Verd_{\boldsymbol{\omega}}$ es $\mathcal{A}$-r. e., por lo cual

$$\mathcal{A}^*-Verd_{\boldsymbol{\omega}}=(\mathcal{A}^*-S^{\tau_A})\cup(S^{\tau_A}-Verd_{\boldsymbol{\omega}})$$

lo es. Es decir que $Verd_{\boldsymbol{\omega}}$ y su complemento son $\mathcal{A}$-r.e. por lo cual $Verd_{\boldsymbol{\omega}}$ es $\mathcal{A}$-r. $\blacksquare$

Ahora podemos probar el importante resultado anunciado.

Proposition 272 *$Verd_{\boldsymbol{\omega}}$ no es $\mathcal{A}$-r.e.*

Proof. Por el Lema 269 hay un predicado $\emptyset$-p.r., $P:\omega\times\omega\rightarrow\omega$ tal que el predicado $Q=\lambda x\left[(\exists t\in\omega)P(t,x)\right]:\omega\rightarrow\omega$ no es $\emptyset$-recursivo. Notese que Q tampoco es $\mathcal{A}$-recursivo. Ya que P es representable, hay una formula $\varphi=_d\varphi(v_1,v_2,v)\in F^{\tau_A}$ la cual cumple

$$\boldsymbol{\omega}\models\varphi\left[t,x,k\right] \text{ si y solo si } P(t,x)=k$$

cualesquiera sean $t,x,k\in\omega$. Sea $\psi=\varphi(v_1,v_2,1)$. Declaremos $\psi=_d\psi(v_1,v_2)$. Tenemos entonces

$$\boldsymbol{\omega}\models\psi\left[t,x\right] \text{ si y solo si } P(t,x)=1$$

cualesquiera sean $t,x\in\omega$. Sea $\psi_0=\exists v_1\ \psi(v_1,v_2)$. Declaremos $\psi_0=_d\psi_0(v_2)$. Tenemos entonces

$$\boldsymbol{\omega}\models\psi_0\left[x\right] \text{ si y solo si } Q(x)=1$$

cualesquiera sea $x\in\omega$. Por el lema de reemplazo tenemos que para $x\in\omega$,

$$\boldsymbol{\omega}\models\psi_0\left[x\right] \text{ si y solo si } \boldsymbol{\omega}\models\psi_0(\widehat{x})$$

(justifique), por lo cual

$$\boldsymbol{\omega}\models\psi_0(\widehat{x}) \text{ si y solo si } Q(x)=1$$

cualesquiera sea $x\in\omega$. Ya que $\psi_0(\widehat{x})$ es una sentencia,

$$\psi_0(\widehat{x})\in Verd_{\boldsymbol{\omega}} \text{ si y solo si } Q(x)=1$$

Sea $h:\omega\rightarrow\mathcal{A}^*$, dada por $h(x)=\psi_0(\widehat{x})$. Es facil ver que h es $\mathcal{A}$-recursiva. Ya que $Q=\chi_{Verd_{\boldsymbol{\omega}}}^{\mathcal{A}^*}\circ h$ y Q no es $\mathcal{A}$-recursivo, tenemos que $\chi_{Verd_{\boldsymbol{\omega}}}^{\mathcal{A}^*}$ no es $\mathcal{A}$-recursiva, es decir que $Verd_{\boldsymbol{\omega}}$ es un conjunto no $\mathcal{A}$-recursivo. El lema anterior nos dice entonces que es $Verd_{\boldsymbol{\omega}}$ no es $\mathcal{A}$-r.e.. $\blacksquare$

Ahora si, estamos en condiciones de probar facilmente el famoso resultado de Godel.

Theorem 273 (Teorema de incompletitud) *Si* $\Sigma \subseteq Verd_{\boldsymbol{\omega}}$ *es* $\mathcal{A}$*-r.e., entonces* $Teo_{(\Sigma,\tau_A)} \subsetneq Verd_{\boldsymbol{\omega}}$

Proof. Ya que $\boldsymbol{\omega}$ es un modelo de (Σ, τ_A), por el Teorema de Correccion, tenemos que $Teo_{(\Sigma,\tau_A)} \subseteq Verd_{\boldsymbol{\omega}}$. Ya que $Teo_{(\Sigma,\tau_A)}$ es $\mathcal{A}$-r.e (Proposicion 266) y $Verd_{\boldsymbol{\omega}}$ no lo es, tenemos que $Teo_{(\Sigma,\tau_A)} \neq Verd_{\boldsymbol{\omega}}$. ∎

Corollary 274 *Existe* $\varphi \in S^{\tau_A}$ *tal que* $Arit \nvdash \varphi$ *y* $Arit \nvdash \neg\varphi$.

Proof. Dejamos al lector la prueba de que el conjunto Σ_A es $\mathcal{A}$-r.e.. Una ves probado esto, podemos aplicar el teorema anterior a la teoria $Arit = (\Sigma_A, \tau_A)$, lo cual nos dice que $Teo_{Arit} \subsetneq Verd_{\boldsymbol{\omega}}$. Sea $\varphi \in Verd_{\boldsymbol{\omega}} - Teo_{Arit}$. O sea que $Arit \nvdash \varphi$ y $\varphi \in Verd_{\boldsymbol{\omega}}$. Ya que $\neg\varphi \notin Verd_{\boldsymbol{\omega}}$, tenemos que $\neg\varphi \notin Teo_{Arit}$, es decir $Arit \nvdash \neg\varphi$. ∎